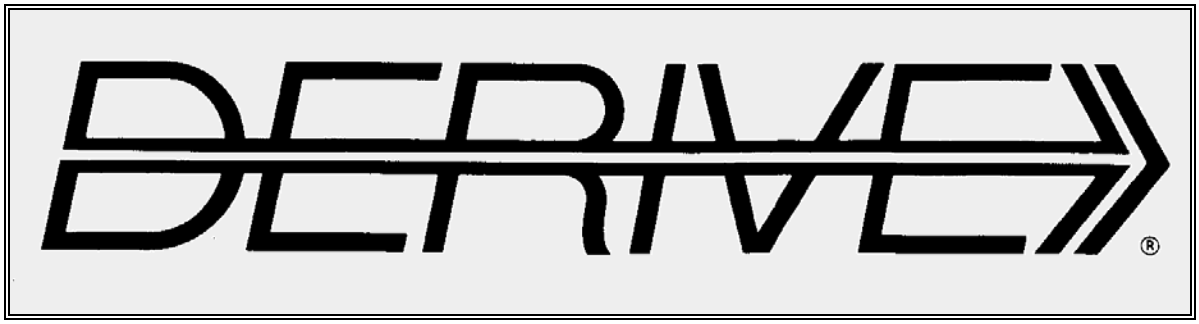
THE DERIVE - NEWSLETTER #58

THE BULLETIN OF THE



USER GROUP + CAS-TI

Contents:

June 2005

Hi Derivers,

The Texas Instruments website has a free, collaborative forum where you can now share your favorite Derive activities and upload Derive worksheets and math files. The online Activities Exchange allows educators to post an activity that has proven effective in class - or browse submissions from colleagues.

For more information on the Activities Exchange, visit

http://education.ti.com/exchange .

From this page, you can browse by subject or TI product; search for specific activities; or submit a new activity. When you click on the "Submit an activity" link you will be asked to log onto the TI web site. (If you don't already have a User Name and Password, simply register to create one.) Afterwards, you will be guided through a series of screens to input the activity information and upload activity file(s). Please complete all the online template components (e.g., "before the activity, "during the activity", "other materials", "after the activity", etc.) as more information encourages activity usage.

Below are links to example Derive activities:

http://education.ti.com/educationportal/activityexchange/activity_detail.do?activityid=4170&cid=us

http://education.ti.com/educationportal/activityexchange/activity_detail.do?activityid=4172&cid=us

Regards,

Theresa Shelby
Texas Instruments

I am very grateful for permission to present one sample lesson in this DNL as a mouthwatering (page 33), Josef

CAME 2005 Symposium: Second Announcement
Shaping Research and Development of Computer Algebra in Mathematics Education

To be held in October 19-20, 2005 in Roanoke, Virginia USA
in conjunction with PME-NA-27, October 20-23, 2005
http://www.pmena.org/2005/

Themes of the Symposium

1. CAS, instrumentation and the anthropological approach

Presenter: John Monaghan, University of Leeds, UK

2. The impact of CAS on our understanding of mathematics education

Presenter: Werner Peschek, University of Klagenfurt, Austria

3. Teacher learning while teaching with CAS

Presenter: Baerbel Barzel, Universität Duisburg-Essen, Germany

Each of the three themes will be addressed in plenary lectures and in topic groups, where the issues will be worked on in more detail, based on the experiences of the participants. The topic groups will aim at producing a synthesis of the discussions in the plenary and topic group sessions. In the discussion groups there will be, apart from time set aside to discuss issues arising from the plenary addresses, opportunity for further short presentations.

The plenary papers will appear on CAME's Website before the Symposium:
http://www.lonklab.ac.uk/came/

Dear DUG members,

When I started collecting stuff for DNL#58 I intended to end up with about 40-44 pages. As you can see it became more than 50 pages. There is one contribution from Rüdeger Baumann not only answering an until now unanswered challenge but also offering a fine introduction in treating strings and in programming with them. Speaking about programming Titbits 30 must be mentioned. Johann Wiesenbauer recommends his Titbits because of containing some examples of very efficient programming. In my opinion Titbits 30 dealing with elliptic curves are one of his very best in his long series of Titbits published in the Newsletters so far, many thanks Johann, outdoing yourself once more.

I am very grateful for permission to reprint Carl Leinbach´s paper on GPS which can be found among the many activities which can be downloaded from the TI-website (Information page). Carl´s contribution is a mouthwatering example standing for many other papers. There is only little experience needed to transfer the activities from PC to handheld and vice versa.

Please notice the many announcements of useful websites and conferences presented in this DNL.

Last weekend I was in the happy position to attend USACAS 3 in Atlanta. There is a wonderful group of enthusiastic US-teachers led by JIm Schultz, Natalie Jakucyn and Bob Mc Collum who want to propagate the use of CAS in US curriculum.

Natalie Jakucyn made it possible to organize a CAS-strand at the 2006 NCTM Regional Conference in Chicago. Please support this great idea by submitting a CAS-related paper (deadline 11 July 2005, see last page!!).

Thanks also to Ernest Carpenter from South Africa – he urged to publish the Coons article – for his mails and the provided websites on "NURB-surfaces".

I wish you a wonderful summer and am looking forward to providing DNL#59 and the revised DNL#7 in fall.
Best regards to you and your families

Some Australian websites: (University of Melbourne)

`extranet.edfac.unimelb.edu.au/DSME/CAS-CAT/`

`extranet.edfac.unimelb.edu.au/DSME/RITEMATHS/`

`edfac.unimelb.edu.au/DSME/`

`www.vcaa.vic.edu.au/vce/studies/mathematics/caspilolt/casindex.html`

2003 PME Proceedings online: `online.terc.edu/`

Canadian website: `www.edu.gov.on.ca`

Individuals on Ontario schools who could provide additional information:
`Thomas_Steinke@occdsb.on.ca` and `fred.ferneyhough@peelsb.com`.

A Derive dedicated website in Belgium: `http://cage.ugent.be/~svw/DfW/`

Download all *DNL-DERIVE*- and TI-files from

`http://www.austromath.at/dug/`

`http://www.bk-teachware.com/main.asp?session=375059`

The *DERIVE-NEWSLETTER* is the Bulletin of the *DERIVE* & CAS-*TI User Group*. It is published at least four times a year with a contents of 44 pages minimum. The goals of the *DNL* are to enable the exchange of experiences made with *DERIVE* and the *TI*-89/92/*Titanium*/*Voyage* 200 as well as to create a group to discuss the possibilities of new methodical and didactical manners in teaching mathematics.

As many of the *DERIVE* Users are also using the CAS-*TIs* the *DNL* tries to combine the applications of these modern technologies.

Contributions:

Please send all contributions to the Editor. Non-English speakers are encouraged to write their contributions in English to reinforce the international touch of the *DNL*. It must be said, though, that non-English articles will be warmly welcomed nonetheless. Your contributions will be edited but not assessed. By submitting articles the author gives his consent for reprinting it in the *DNL*. The more contributions you will send, the more lively and richer in contents the *DERIVE* & CAS-*TI Newsletter* will be.

Editor: Mag. Josef Böhm
A-3042 Würmla
D´Lust 1
Austria
Phone/FAX: 43-(0)2275/8207
e-mail: nojo.boehm@pgv.at

Next issue: September 2005
Deadline 15 August 2005

Preview: Contributions waiting to be published (selection)

Pringles, B. Grabinger, GER
Two Stage Least Squares, M. R. Phillips, USA
Some simulations of Random Experiments, J. Böhm, AUT & L. Kopp, GER
Wonderful World of Pedal Curves, J. Böhm
Another Task for End Examination, J. Lechner, AUT
Tools for 3D-Problems, P. Lüke-Rosendahl, GER
ANOVA with *DERIVE* & *TI*, M. R. Phillips, USA
Hill-Encription, J. Böhm
Farey Sequences on the *TI,* M. Lesmes-Acosta, COL
Simulating a Graphing Calculator in *DERIVE*, J. Böhm, AUT
Henon & Co, J. Böhm
Challenges from Fermat, Bj. Felsager, DEN
Are all Bodies falling equally fast, J. Lechner, AUT
Modelling Traffic Density, Th. Himmelbauer, AUT
Do you know this? Cabri & CAS on PC and Handheld, W. Wegscheider, AUT
An Interesting Problem with a Triangle, P. Lüke-Rosendahl, GER
Diophantine Polynomials, Duncan McDougall, CAN
Rosettes, J. Lechner, AUT
Jacobi-Iteration, Heinz Rainer Geyer, GER

and Setif, FRA; Vermeylen, BEL; Leinbach, USA; Koller, AUT; Baumann, GER; Keunecke, GER, ...and others

Impressum:
Medieninhaber: *DERIVE* User Group, A-3042 Würmla, D´Lust 1, AUSTRIA
Richtung: Fachzeitschrift
Herausgeber: Mag.Josef Böhm
Herstellung: Selbstverlag

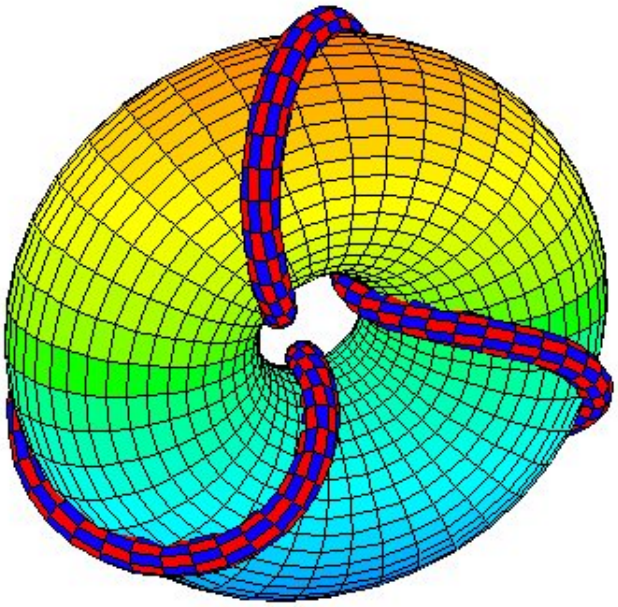
Steve Schonefeld sschonefeld@mchsi.com

Hi Joseph & Noor.

Every time I look at my poster from the Montreal DERIVE/TI conference, I remember the folks in charge of that conference confiding that the graphics for that poster were not created using DERIVE. They were created using MAPLE. I hope we can remedy that for the next conference. DERIVE can create many beautiful and interesting graphics. Perhaps members of DUG can contribute graphics for the next poster. Let me start the ball rolling with the attached contribution. It is a torus wrapped in a torus knot.

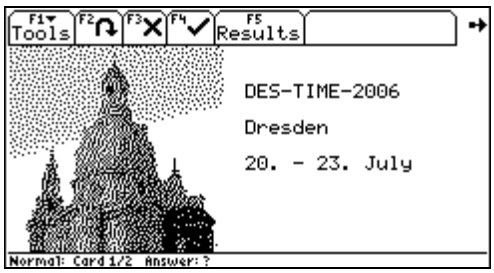
Wishing you and yours a happy Easter.

Steve Schonefeld



Great idea!

So let´s open a contest for the DES-TIME 2006 poster.



DES-TIME 2006 in Dresden, Juli 20 – 23, 2006

www.des-time-2006.de

Ignacio Larrosa Cañestro ilarrosa@mundo-r.com

I define the function s(n) and then I simplify to obtain a list for s(n):

$$s(n) := \left(\sum_{i=1}^{n} \text{DIVISOR\_TAU}(i) \right) - n$$

```
VECTOR(s(n), n, 95, 105)
```

[352, 363, 364, 369, 374, 382, 383, 102, 103, 104, 105]

But it's correct only until n = 101, s(101) = 383. Above n = 101, I get incomprensibly s(n) = n ...

Note that `s(n)` must be strictally increasing, because `DIVISOR_TAU(n)` > 1 if n > 1. And `DIVISOR_TAU(i)` works well for any of that values.
(`DIVISOR_TAU(n)` = number of divisors of n.)

I try it with Derive 6.10 and Derive 5.06, getting exactly the same results. I don't understand anything.

Best regards,

Ignacio Larrosa Cañestro
A Coruña (España)

Albert D. Rich

Hello Ignacio,

If the difference between the upper and lower limits of a definite sum is greater than 100, Derive attempts to compute the sum by finding an anti-difference and substituting in the limits. This is analogous to using anti-derivatives to compute definite integrals. See Derive's on-line help on summation for details.

This is the reason for the change in behavior of s(n) for n>101. Unfortunately, the anti-difference for `DIVISOR_TAU(i)` is not computed correctly because of calls on special functions like `FACTORS`.

Until this problem is resolved in a future release of Derive, you can force Derive to directly compute definite sums using iteration instead of trying to find an anti-difference by specifying a step size of 1. Thus if s(n) is defined as

```
s(n) := SUM(DIVISOR_TAU(i), i, 1, n, 1) - n
```

instead of as

```
s(n) := SUM(DIVISOR_TAU(i), i, 1, n) - n
```

then s(102) simplifies to 390.

Hope this helps.

Aloha,
Albert D. Rich
Co-author of Derive

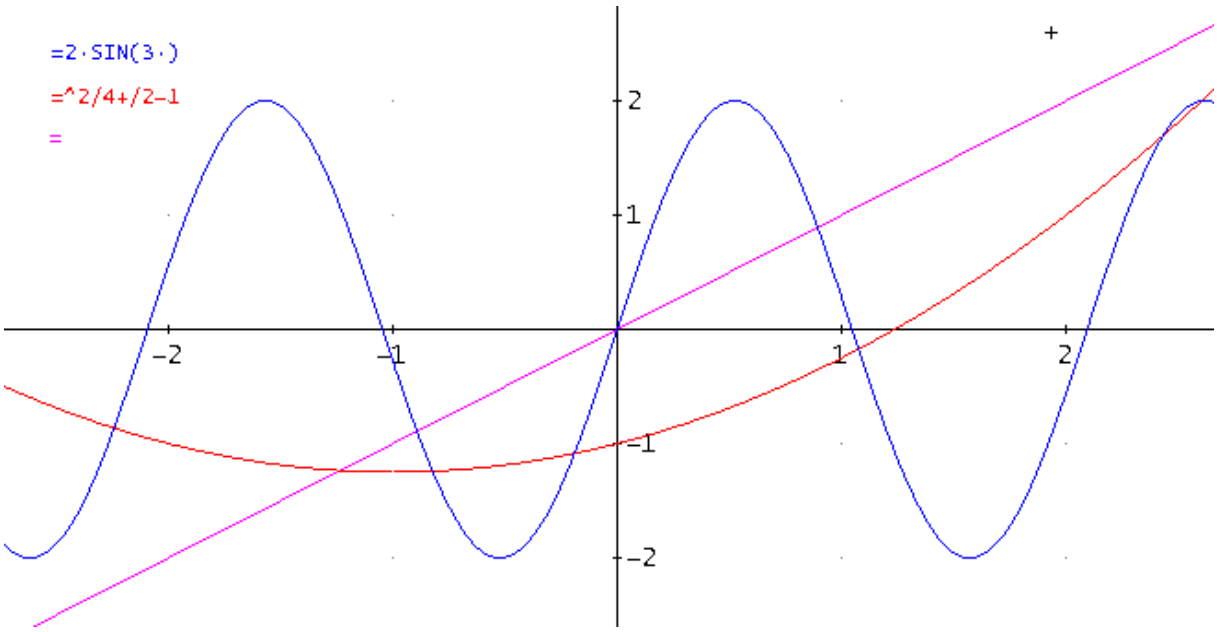
```
s(n) := Σ(DIVISOR_TAU(i), i, 1, n, 1) - n

VECTOR(s(n), n, 95, 105)

[352, 363, 364, 369, 374, 382, 383, 390, 391, 398, 405]
```

Tania Koller, St.Pölten, Austria

Some of Tania´s students had problems with annotating new plots in the 2D Plot Window.
I came across this strange behaviour, too and I could not find the reason for its occurrence.

I wanted to plot y = x, y = x^2/4+x/2-1 and f(x):=2 sin(3x) with annotating the graphs, but the variables didn´t appear as shown on the screen shot.

Any idea what happened??



Theresa Shelby

Thank you very much for the information and the file. The mystery is solved!

After the annotation is created, Derive substitutes the Horizontal Axis Title for every occurrence of x and the Vertical Axis Title for every occurrence of y in the annotation of a plotted expression. Currently, if a title in the plot window's Options > Display > Axes is blank, then an empty string will be substituted into the annotation. So, in this case, if both the horizontal and vertical axis titles are blank, "y=x" becomes "=" and "y = x^2/4+x/2-1" becomes "=^2/4+/2-1".

In order to keep the horizontal and vertical titles invisible, yet still preserving the annotations, I advise using the Options > Display > Axes tab to restore the default titles of "x" and "y" and choose a font color for the axes titles that's close enough to the background of the plot window (e.g., "white") so as not to be visible.

For the next release of Derive, I will modify the code so that it does no substitution of the Axes Titles in the annotation if the title is empty.

Aloha,
Theresa

Status Variables saved?

A question from a colleague: Why is it not possible in DERIVE 6 to save status variables like Notation Digits, Output Decimal, ... as it was in Derive 5. Is there any way to do this.

My advice was to create an INI.MTH containing all requested settings and then load this INI.MTH as a utility file. Is there a better way to modify the INI-file??

Second question: Preparing a presentation I came across that DERIVE 6 behaves strange in performing simplifications. I attach parab.dfw. `ut(p1,p2)` is the DERIVE 5 version giving the correct result in DERIVE 5, but giving a wrong simplification in DERIVE 6. It needs to rewrite the function as a program, then it works properly in all cases. (`ut_v6(p1,p2,u)`).

```
ut(p1, p2) :=
  If ABS(x) ≤ 5 + p1/10
     5·(60 - p2)/(p1 + 50)^2·x^2 + (p2 - 60)/20
     ?
```

$$ut(0,\ 0) := IF\left(-5 \le x \le 5,\ \frac{x^2}{500} - 3,\ ?\right)$$

```
ut_v6(p1, p2, u) :=
  Prog
    u := 5·(60 - p2)/(p1 + 50)^2·x^2 + (p2 - 60)/20
    If ABS(x) ≤ 5 + p1/10
       u
       ?
```

$$ut\_v6(0,\ 0) := IF\left(-5 \le x \le 5,\ \frac{3\cdot x^2}{25} - 3,\ ?\right)$$

expr #2 is not correct, expr #4 is correct!

$$ut(1,\ 1) := IF\left(-\frac{51}{10} \le x \le \frac{51}{10},\ \frac{5\cdot x^2\cdot(60-1)}{(1+50)^2} - \frac{59}{20},\ ?\right)$$

$$ut\_v6(1,\ 1) := IF\left(-\frac{51}{10} \le x \le \frac{51}{10},\ \frac{295\cdot x^2}{2601} - \frac{59}{20},\ ?\right)$$

both versions give the correct answer!

Best regards
Josef

Albert Rich

Your first question is answered in the Derive 6.10 on-line under the User Interface Questions of the Frequently Asked Questions and Answers section. They are copied here:

In order to restore the environment in effect at the time worksheets are saved, the expressions in the algebra window, as well as all the user-defined functions and variable assignments, are saved in dfw files. In earlier versions of Derive, the values of state variables (for example, the precision; notation; radix base; and the exponential, log, and trig simplification settings) were not saved in dfw files. This made it impossible to fully restore the worksheet's environment when it was opened. Therefore, Derive 6 saves the state variable values in dfw files (for details, see Using and Saving State Variables). So now if while working at 20 digits of precision you save a worksheet, that same precision is restored when the worksheet is re-opened.

When earlier versions of Derive started and a new blank worksheet created, the state variable values were set to those in effect the last time Derive was shutdown. These settings may have long since been forgotten by the user and/or not appropriate for the task at hand. Therefore, to avoid confusion, when Derive 6 starts the state variable values are set to their factory defaults. So there is no need to save state variable values in the initialization file. However, Derive 6 continues to save the Windows state information (for example, window sizes and colors, printing options, etc.) in the file (for details, see the Derive Initialization File).

Your second question: This unfortunate bug occurs only in version 6.10 of Derive 6. It has recently been found and corrected for the next release of Derive. I apologize for the inconvenience.

Aloha,
Albert

Ernest Carpenter (South Africa)

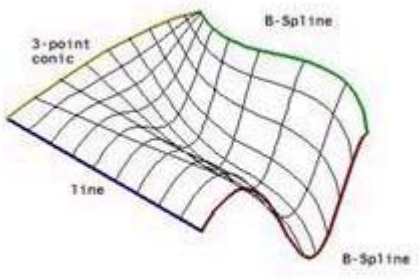
Hi Josef,

Thanks for the files relating to the Coons-Surfaces. I enjoyed it tremendously (especially after it was translated☺) for mainly the following reasons:

- Firstly it was clear and nicely presented and made a topic, that usually make me feel more math-illiterate than I'm currently is due to the sometimes complex representation in literature, a little bit more understanding!
- Working with CAD (Microstation, AutoCAD, etc.) and modeling systems (Amapi, Maya, 3D Max 5, Cinema 4D, etc.), I constantly use these NURB related surfaces without always fully understanding the main principles behind the construction – now it makes more sense!
- It prompted me to 'research' the tools I use and get a better understanding of how big and complex curve and surface modeling are! (Interesting history - http://rt001kxn.eresmas.net/principal/history_CAGD.pdf)

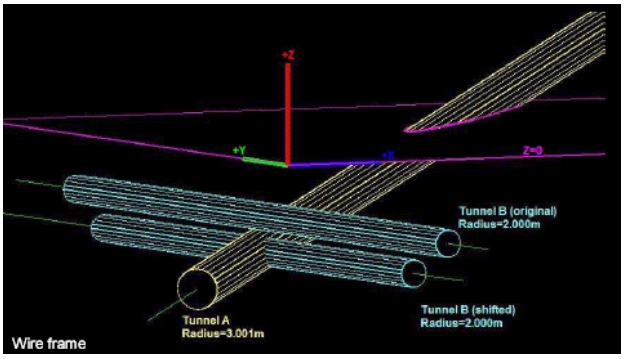
(one of the packages I also use is Eovia's Amapi and a nice tutorial showing how some of these surfaces are used in modeling is: http://server2.eovia.com/tutorials/amapi6/shoe/)

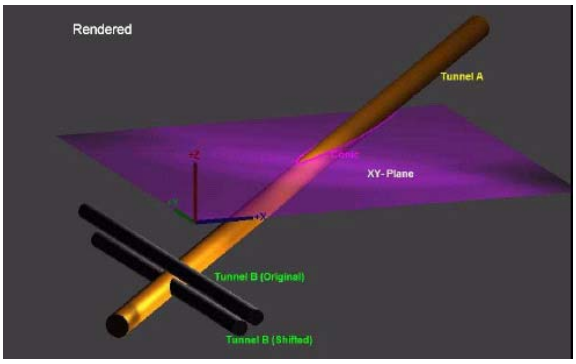
Right is a quick CAD representation of a NURB surface using a degenerated Coons-Gordon algorithm (Microstation)

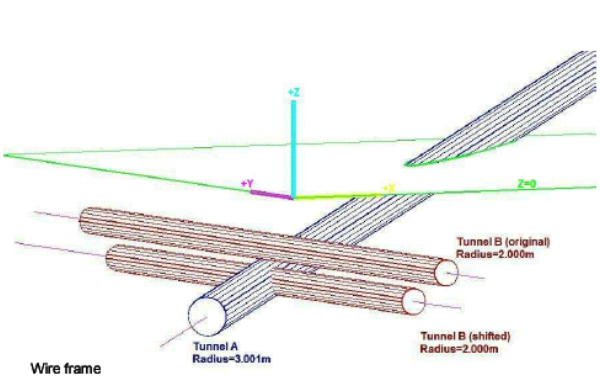


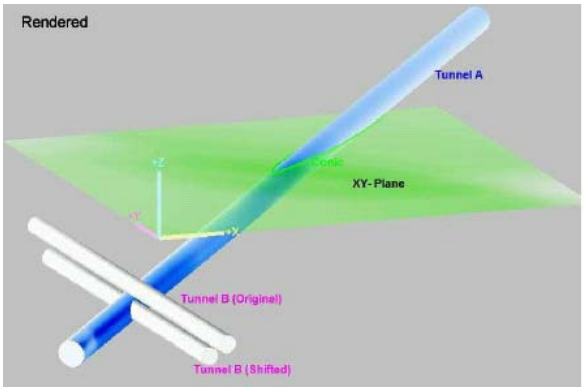
I also immensely enjoyed the new and revised DNL's.

As an exercise I 'modeled' the "Two Underground Tunnels" problem in CAD to see how the geometric results compare with the numerical results in DNL05. They are the same☺!









Rüdeger submitted this contribution some time ago, but this does not influence its interesting nature. His article isiwritten in german and I will try to provide short English summaries. In the original paper all functions, variables and programs have German names. To make it easier for our non-German speaking members I use English names. There are two DERIVE-files which can be downloaded: engel.dfw with the German names and engel_eng.dfw with the English names. Most of the problems and solutions are self explanatory. I added the respective solutions for our many TI-CAS users. Josef

Engelfolgen und Schieberegister

Zum Umgang mit Zeichenketten und Zahlen

Engel Sequences and Shift Registers
How to treat Strings and Numbers

Rüdeger Baumann, Celle

Wie viele Anfragen und insbesondere auch die Leserzuschriften in DNL 49 zeigen, besteht nach wie vor Bedarf an elementaren Einführungsaufgaben und Übungen zum Programmieren in Derive. Zwar hat Josefs Buch hier Abhilfe geschaffen – aber andererseits sind natürlich noch Fragen offen geblieben. Im folgenden sollen einige elementare Überlegungen zur Behandlung von Zeichenketten und Zahlen angestellt werden, die erfahrungsgemäß dem Anfänger Schwierigkeiten bereitet. Als Anwendung werden zwei interessante Aufgaben gelöst.

There are many requests on introductory examples and exercises in programming in Derive. In this contribution some elementary considerations on the treatise of strings and numbers are presented. Then using the acquired knowledge two interesting problems will be solved.

Elementare Operationen

Zeichenketten (engl.: strings), also (aus Buchstaben, Satzzeichen, Ziffern usw. gebildete) Wörter und Sätze werden zwischen obenstehende „Gänsefüßchen“ (engl.: double quotes) eingeschlossen. Die Operationen für Zeichenketten und Listen (in Derive: Vektoren) sind eng miteinander verwandt: man kann die Listenoperationen auch auf Wörter, Zahlen und Texte anwenden. Die drei Beispiele behandeln die wechselseitige Umwandlung zwischen Zahlen, Zeichenketten und Listen.

Beispiel 1: Eine gegebene Zahl soll in eine Zeichenkette verwandelt werden – und umgekehrt.
Example 1: A given number shall be transformed into a string – and vice versa.

Die Aufgabe *Zahl in Wort* (*NumberToWord*) lässt sich sehr einfach mit Hilfe der Funktion `STRING` lösen:

#1: `NumberToWord(n) := STRING(n)`

#2: `NumberToWord(3.14159) = [/, 314159, 100000]`

`["-", 314159/100000]`

#3: `NumberToWord(-3.14159) =` $\left[-, \frac{314159}{100000}\right]$

This is the way, the result appears in Exact Simplification.

#4: `NumberToWord(3.14159)`

Approximate #4 and #6

#5: 3.14159

#6: NumberToWord(-3.14159)

#7: [-, 3.14159]

#8: NAME_TO_CODES(-3.14159) = [45, 51, 49, 52, 49, 53, 57, 47, 49, 48, 48, 48, 48, 48]

#9: NumberToWord2(n) := APPROX(CODES_TO_NAME(NAME_TO_CODES(n)))

#10: NumberToWord2(-3.14159) = -3.14159

Die Umkehrung *Wort in Zahl* ist nicht ganz so einfach, denn ein Wort kann Zeichen enthalten derart, dass es sich nicht als Zahl interpretieren lässt. Wir wandeln das Zahlwort zunächst – mit Hilfe der Funktion NAME_TO_CODES – in eine Liste von ASCII-Nummern um:
The reverse operation *WordToNumber* is not so easy, because there might be characters which cannot be interpreted as a number. At first we transform the number-word into a list of ASCII-Codes.

#11: WordToASCIIList(w) := NAME_TO_CODES(w)

#12: WordToASCIIList(-3140) = [45, 51, 49, 52, 48]

#13: WordToASCIIList(3.14) = [49, 53, 55, 47, 53, 48]

Auf diese Liste wenden wir die Funktion CODES_TO_NAME an, die aus einer Liste von ASCII-Nummern eine Zahl macht. Sie weist allerdings Besonderheiten bezüglich der Null auf:

CODES_TO_NAME([48]) = 0
CODES_TO_NAME([48, 51]) = 3

Das heißt, die vordere Null wurde weggelassen. Aus diesem Verhalten entstehen zuweilen Komplikationen. Insgesamt erhalten wir

#14: WordToNumber(w) := CODES_TO_NAME(NAME_TO_CODES(w))

#15: WordToNumber(0123) = 123

#16: WordToNumber(-1.23) = -1.23

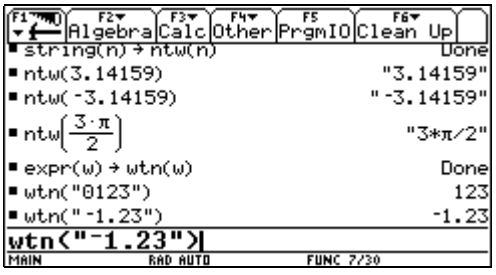
Das Argument ist ein String, daher unter " zu schreiben.

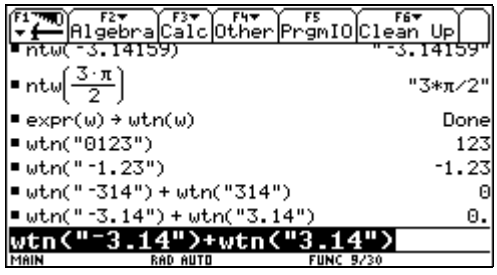
As you can see the leading zero is omitted. This might cause sometimes problems. The argument is a string and must be written under ". It works with negative integers, but not with negative decimal numbers.

#17: WordToNumber(-314) + WordToNumber(314) = 0

#18: WordToNumber(-3.14) + WordToNumber(3.14) = -3.14 + 3.14

The CAS-TIs provide two functions `string(number)` and `expr(string)` for these purposes:





Beispiel 2: Einer natürlichen Zahl soll die Liste ihrer Ziffern zugeordnet werden – und umgekehrt.
Example 2: A natural number shall be assigned to the list of its digits – and vice versa.

Bei einer Zahl n kann man die i-te Ziffer (von links) durch `n SUB i` (bzw. `n↓i`) herausgreifen:

```
      NumberToList(n) := VECTOR(n     , i, DIM(n))
#19:                              i
#20:  NumberToList(3141590) = [3, 1, 4, 1, 5, 9, 0]
#21:  NumberToList2(n) := VECTOR(k - 48, k, NAME_TO_CODES(n))
#22:  NumberToList2(3141590) = [3, 1, 4, 1, 5, 9, 0]
```

Even entering `NumberToList(03141590)` the leading zero does not appear in the Algebra Window.

Die Umkehrung *Liste in Zahl* geht wie folgt: Wir definieren eine Funktion `asc` durch

```
#23:  asc(z) := FIRST(NAME_TO_CODES(z))
#24:  asc(3) = 51
```

Die Funktion `NAME_TO_CODES` liefert eine Liste; ihr entnehmen wir mittels `FIRST` das erste (und einzige) Element.

```
#25:  ListToNumber(v) := CODES_TO_NAME(VECTOR(FIRST(NAME_TO_CODES(k)), k, v))
#26:  ListToNumber([0, 3, 1, 4, 0]) = 3140
```

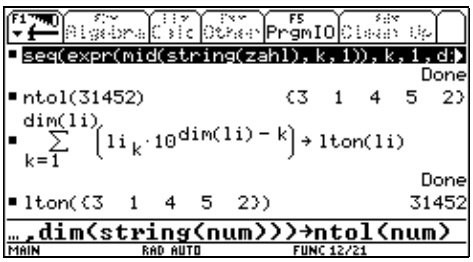
Es gibt auch eine rein arithmetische Lösung:

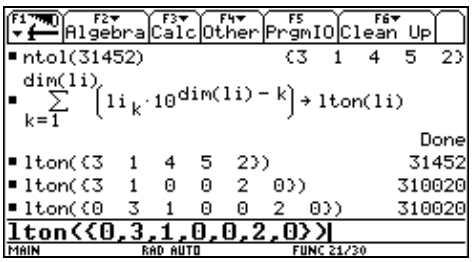
```
      NumberToList1(n) :=
        If n ≤ 0
#27:       []
           ADJOIN(MOD(n, 10), NumberToList1(FLOOR(n, 10)))
#28:  NumberToList1(3141590) = [0, 9, 5, 1, 4, 1, 3]
```

Die Ziffern werden in umgekehrter Reihenfolge aufgelistet! Dies kann manchmal nützlich sein. Wünschen wir uns die Ziffern in normaler Reihenfolge, hängen wir sie mittels `APPEND` hinten an. Da diese Funktion Listen verkettet, machen wir aus der Ziffer `MOD(n,10)` durch Einschließen in eckige Klammern eine (einelementige) Liste. Die Funktion lautet:

```
      NumberToList2(n) :=
        If n ≤ 0
#29:       []
           APPEND(NumberToList2(FLOOR(n, 10)), [MOD(n, 10)])
#30:  NumberToList2(3141590) = [3, 1, 4, 1, 5, 9, 0]
```

`NumberToList1` returns the digits in reverse order which can be useful sometimes. To obtain the regular order we use `APPEND` instead of `ADJOIN`. Take care for the brackets enclosing the digit `MOD(n,10)` in `NumberToList2` making this digit to a list containing only one element. This is necessary, because `APPEND` works with lists only.





Beispiel 3: Einer gegebenen Zeichenkette soll die Liste ihrer Zeichen zugeordnet werden – und umgekehrt.

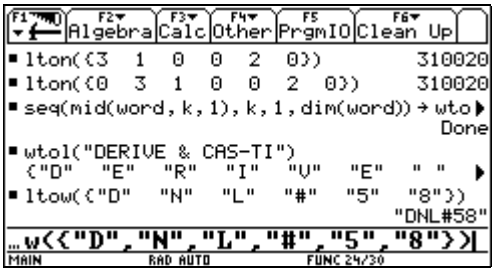
Example 3: A given string shall be assigned to the list of its charcters – and vice versa.

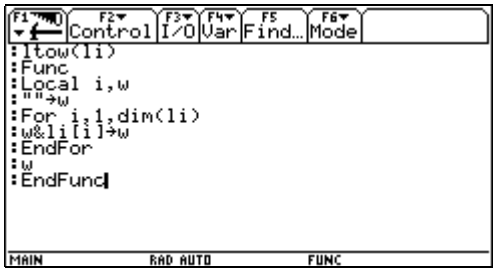
Die Aufgabe *Wort in Liste* löst sich analog zu Beispiel 2:

```
#31:  WordToList(w) := VECTOR(w , i, DIM(w))
                              i

#32:  WordToList(DERIVE & CAS-TI) = [D, E, R, I, V, E,  , &,  , C, A, S, -, T, I]
```

Für die Umkehrung *Liste in Wort* orientieren wir uns ebenfalls an Beispiel 2:
`ListToWord0(v)` will resolve the problem with the omitted leading zero(s).

```
#33:  ListToWord(v) := CODES_TO_NAME(APPEND(NAME_TO_CODES(v)))

#34:  ListToWord([D, N, L, #, 5, 8]) = DNL#58
```





Die Engelfolge / The Engel Sequence

In der Zeitschrift *Praxis der Mathematik* stellte Arthur Engel im Jahr 1967 folgende Aufgabe:
1967 Arthur Engel posed a problem in *Praxis der Mathematik*:

> „Es werden die zehn Ziffern $z_1, z_2, ..., z_{10}$ beliebig angenommen. Diese Ziffernfolge wird nach dem Gesetz $z_n = z_{n-10} + z_{n-9}$ (mod 10) mit n = 11, 12, 13, ... fortgesetzt. Man zeige, dass die Folge periodisch wird, ja sogar sofortperiodisch. Was kann man über die Periodenlänge aussagen?“
>
> Let $z_1, z_2, ..., z_{10}$ ten arbitrary digits. This sequence is continued according to the following rule: $z_n = z_{n-10} + z_{n-9}$ (mod 10) with n = 11, 12, 13. Show that this sequence is periodical and that there is no preperiod appearing.
>
> Beispiel/*Example*: 1234567890 3579135793826048...

Wir wollen hier – als Anwendung der Überlegungen zu Zeichenketten und Zahlen – die Erzeugung der Folge programmieren. Zunächst hat man sich zu entscheiden, ob die Folge als Liste, Zahl oder Zeichenkette geschrieben werden soll. Es handelt sich um Ziffern, die modulo 10 addiert werden: dies spricht für die Verwendung von Zahlen. Andererseits müssten wir dann auf weit zurückliegende Ziffern einer (großen) Zahl zugreifen, die Zahl um eine Stelle verlängern und die neue Ziffern als Einerziffer anfügen: dies scheinen schwierige arithmetische Operationen zu sein. Somit empfiehlt sich die Verwendung von Zeichenketten.

Instead of working with numbers we recommend the use of strings because we have to refer to numbers lying far back and form new numbers. This seems to cause difficult arithmetic operations.

Wir stellen zunächst die Länge k des gegebenen Worts (Ziffernfolge) fest und bauen dann eine Schleife (LOOP) auf, die zur Bestimmung der i-ten Ziffer (i > k) um k bzw. um k – 1 Schritte zurückgreift: Folge `SUB (i - k)` bzw. Folge `SUB (i - k + 1)`. Da es sich um Zeichen handelt, müssen sie in Zahlen umgewandelt werden (siehe oben), die sich dann modulo 10 addieren lassen und rechts angehängt werden. Hier wirkt die Funktion APPEND auf Zeichenketten (von Derive als Listen aufgefasst).

Um die aus der Folge herausgegriffenen Ziffern als Zahlen behandeln zu können, müssen wir die Aufgabe *Ziffer in Zahl* lösen:

As we have to treat the digits as numbers we have to solve the problem DigitToNumber.

```
#37:  DigitToNumber(z) := FIRST(NAME_TO_CODES(z)) - 48

#38:  DigitToNumber(3) = 3
```

Zuerst wird der Ziffer (hier: 3) ihre ASCII-Nummer (51) zugeordnet und sodann die der Null (48) abgezogen: 51 – 48 = 3.

The program `arthur(start, length)` with first results:

```
      Arthur(start := "1234567890", length := 100, seq, z1, z2) :=
        Prog
          seq := start
          k := DIM(start)
          i := k
          Loop
#39:        i :+ 1
            If i > length
               RETURN seq
            z1 := DigitToNumber(seq↓(i - k))
            z2 := DigitToNumber(seq↓(i - k + 1))
            seq := APPEND(seq, STRING(MOD(z1 + z2, 10)))

#40:  Arthur()

#41:  12345678903579135793826048262108642088318406286149246804753160484128476422530213064783234360151557 79

#42:  Arthur(862, 20) = 86248624862486248624

#43:  Arthur(555, 20) = 55500505550050555005

#44:  Arthur(268, 50) = 26884620828008088866420626884620828008088866420626

#45:  Arthur(220, 50) = 22042466026288068644084822042466026288068644084822

#46:  Arthur(5000, 50) = 50005005505055550005005505055550005005505055550005
```

Can you find the periods?

```
Arthur(862, 20) = 86248624862486248624
Arthur(555, 20) = 55500505550050555005
Arthur(268, 50) = 26884620828008088866420626884620828008088866420626
Arthur(220, 50) = 22042466026288068644084822042466026288068644084822
Arthur(5000, 50) = 50005005505055550005005505055550005005505055550005
```

Wir finden bei dreistelligen Anfangswörtern die Periodenlängen 4 (1), 7 (1), 24 (5), 28 (1), 168 (5); in Klammern steht jeweils die Anzahl der Zyklen. Probe: 1 · 4 + 1 · 7 + 5 · 24 + 1 · 28 + 5 · 168 = 999. Die Periodizität mittels Programm zu erkennen, scheint schwierig. Leider ist das in DNL#47, S. 38 gestellte Problem der Erkennung von Perioden noch nicht gelöst.

Rüdeger notes that the problem to detect periods has not been solved until now (posed in DNL#47). This was reason enough for me to make a try:

```
periode?(num, per, k, i, j) :=
  Prog
    k := 1
    Loop
      If k = DIM(num)
         RETURN "no period detected!"
      per := num↓[1, ..., k]
#47:  i := k + 1
      Loop
        j := IF(MOD(i, k) = 0, k, MOD(i, k))
        If num↓i ≠ per↓j exit
        i :+ 1
        If i > DIM(num)
           RETURN per
      k :+ 1
```

```
#48:  periode?(Arthur(5000, 50)) = 500050055050555
```

```
#49:  DIM(periode?(Arthur(220, 50))) = 24
```

This program works, but to keep it more general it would be fine to also find out if there is a preperiodical part existing. So look at the following:

```
period?(num, per, k, i, j, v, num_) :=
  Prog
    num_ := num
    v := 0
    Loop
      v :+ 1
      k := 1
      Loop
        If k = DIM(num) exit
        per := num↓[1, ..., k]
#50:    i := k + 1
        Loop
          j := IF(MOD(i, k) = 0, k, MOD(i, k))
          If num↓i ≠ per↓j exit
          i :+ 1
          If i > DIM(num)
             RETURN ["Preperiod", "Period"; num_↓[1, ..., v - 1], per]
        k :+ 1
      num := REST(num)
      If DIM(num) = 1
         RETURN "no period detected!"
```

```
#51:  period?(Arthur(268, 50)) = [ Preperiod          Period          ]
                                 [                                    ]
                                 [              26884620828008088664206 ]
```

Some other examples to illustrate `period?(string)`:

```
#54:    5
      ─────
      173
```

```
#55:  0.028901734104046242774566473988439306358381502890173
```

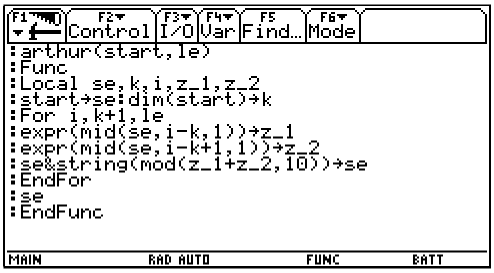
```
#56:  period?(028901734104046242774566473988439306358381502890173)
```

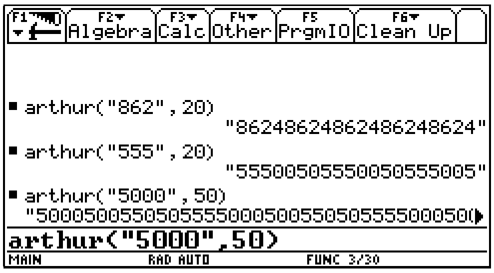
```
#57:  [ Preperiod                    Period                    ]
      [                                                       ]
      [            028901734104046242774566473988439306358381 5 ]
```

```
#58:  period?(002518759018759018759018759018759018759018759)

#59:  [ Preperiod  Period ]
      [   0025     187590 ]

#60:  period?(ABABCABCABC) = [ Preperiod  Period ]
                             [    AB       ABC   ]

#61:  period?(ABCABCDABCDAC) = [ Preperiod    Period   ]
                               [    AB      CABCDABCDA ]

#62:  period?(ABCABCDABCDAE) = no period detected!

#63:  period?(1234526884620828008088866420626884620828008088866420626)

#64:  [ Preperiod            Period          ]
      [   12345    268846208280080888664206  ]

#65:  period?(1234526884620828008088866420626884620828008088866420627)

#66:  no period detected!

      DIM((period?(Arthur(123, 500)))    ) = 168
#69:                                 2,2

      DIM((period?(Arthur(1234, 2000)))    ) = 1560
#70:                                   2,2
```

We can finally count all appearing periodics. #119 - #123 shows the end of this procedure. Rüdeger started with "001" giving 999 three digit starting numbers. I start with "000" resulting in 1000 Engel-Sequences with a three digit initialisation number.

```
#119: p9 := VECTOR(DIM(periode?(Arthur(v, 200))), v, VECTOR(STRING(k), k, 700, 999))

#120: total := APPEND(p1, p2, p3, p4, p5, p6, p7, p8, p9)

#121: [[1, 4, 7, 24, 28, 168], VECTOR(DIM(SELECT(x = v, v, total)), x, [1, 4, 7, 24, 28, 168])]

#122: [ 1  4  7  24   28  168 ]
      [ 1  4  7  120  28  840 ]

#123: Σ([1, 4, 7, 120, 28, 840]) = 1000
```





I´ll leave the program for period detecting with the TI as a challenge for the reader.

Die Erzeugung der Folge können wir uns mittels eines sogenannten Schieberegisters vorstellen (siehe Engel 1991, S. 105 ff). Ein *Schieberegister* (engl.: shift register) besteht aus einer Aneinanderreihung von Speicherzellen, die so geschaltet sind, dass – auf einen Taktimpuls hin – der Inhalt jeder Zelle an die linke Nachbarzelle weitergereicht wird. Dazu gehört eine *Rückkopplungsschaltung*, deren Eingänge mit bestimmten Zellen verbunden sind, und deren Ausgang in den Eingang des Schieberegisters mündet. Die zugehörige (Pseudo-) Zufallsfolge wird von den Zahlen gebildet, die aus der Zelle am linken Rand heraustreten.

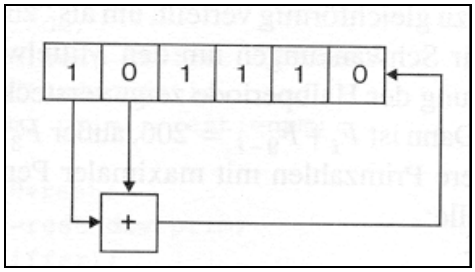


Bild1/Figure 1
Shift Register

We can imagine creating the sequence using a so called shift register (Engel, 1991). This consists of a sequence of cells, which are connected in such a way that at each clock pulse the content of each cell is transmitted to its left neighbour. A reaction coupling is connected with certain cells as input and gives an output which feeds the sequence from the right side. The respective sequence of (pseudo) random numbers is formed by the numbers which leave the register at its left end.

Die einfachste Version ist ein linear rückgekoppeltes Schieberegister (engl.: linear feedback shift register): hier besteht die Rückkopplungsschaltung aus einer XOR-Verknüpfung (Exklusiv-Oder) der angezapften Zellen, was mathematisch der Addition modulo 2 entspricht.

Die Folge der angezapften Zellen, das *Anzapfmuster* (engl.: tap sequence), stellen wir durch einen Null-Eins-Vektor dar (in Bild 1 wäre dies [1, 1, 0, 0, 0, 0]). Dann kann das neue Bit mittels skalarer Multiplikation (in Bild 1: MOD([1, 0, 1, 1, 1, 0] · [1, 1, 0, 0, 0, 0], 2) = 1) bestimmt werden.

The simplest version is a *linear feedback shift register* (LFSR) whose input is the XOR-operation (exclusive or) of some of its outputs. These output cells are called *taps*. The tap sequence is represented by a binary vector. Here it is [1,1,0,0,0,0] because the first two cells are connected by XOR giving the next cell which is here 1 XOR 0 = 1. This new bit can be calculated by a dotproduct modulo 2 of the register content vector and the tap sequence vector MOD([1, 0, 1, 1, 1, 0] · [1, 1, 0, 0, 0, 0], 2) = 1.

The following program for producing a (pseudo) random binary sequence uses working with strings. Let´s assume that we have already produced the sequence 10111011001 and we want to generate bit #12 (i = 12). We need a "window" [6, 7, 8, 9, 10, 11] which selects the last k = 6 bits.

Das folgende Programm macht von einer wichtigen Möglichkeit der Bearbeitung von Zeichenketten Gebrauch: man kann aus einer Zeichenkette durch Vorgabe einer Liste mittels der Operation `SUB` gezielt Elemente herausgreifen. Es sei beispielsweise die Folge 10111011001 bereits gewonnen, und nun soll das zwölfte Bit erzeugt werden (i = 12). Dazu bilden wir das „Fenster“ [6, 7, 8, 9, 10, 11], mit dessen Hilfe die k = 6 letzten Bits herausgegriffen werden:

```
#74:  sequence := [1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1]

#75:  [k := 6, i := 12]

#76:  window := VECTOR(i - k + j, j, 0, k - 1)

      sequence       = [0, 1, 1, 0, 0, 1]
#77:          window
```

Werden sie mit dem Anzapfmuster skalar multipliziert, ergibt sich das neue, in den Eingang zurückgeführte Bit:

```
#78:  pattern := [1, 1, 0, 0, 0, 0]

      MOD(pattern·sequence       , 2) = 1
#79:                      window
```

Damit bekommen wir das folgende Programm:

```
#80:  DigitToNumber(z) := FIRST(NAME_TO_CODES(z)) - 48

      WordToList2(w) := VECTOR(DigitToNumber(w ), i, DIM(w))
#81:                                            i

      ListToWord0(v, w) :=
        Prog
          w := ""
          Loop
#82:        If v = []
              RETURN w
            w := APPEND(w, FIRST(v))
            v := REST(v)

      sr(start := "101110", pattern := "110000", length := 69, seq, pat, window, i, k) :=
        Prog
          seq := WordToList2(start)
          pat := WordToList2(pattern)
          k := DIM(start)
          i := k
#83:      Loop
            i :+ 1
            If i > length
              RETURN ListToWord0(seq)
            window := VECTOR(i - k + j, j, 0, k - 1)
            seq := APPEND(seq, [MOD(pat·seq↓window, 2)])

#84:  sr()

#85:  10111011001101010111111000001000011000101001111010001110010010110111 0
```

We investigate the periodic behaviour:

```
#86:  period?(sr())

      [ Preperiod                                  Period                                  ]
#87:  [                                                                                    ]
      [            10111011001101010111111000001000011000101001111010001110010010110 1 ]

      DIM((period?(sr()))    ) = 63
#88:                    2,2
```

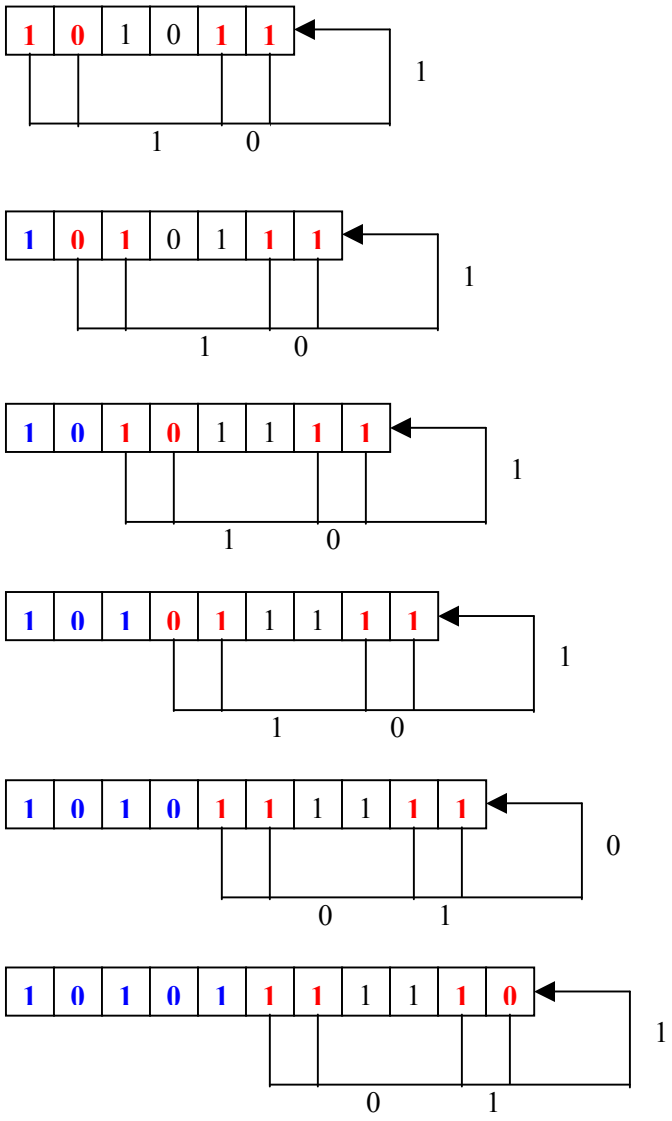
Das Schieberegister liefert die maximale Periodenlänge $2^6 - 1 = 63$. Die Erzeugung von Zufallszahlen mittels Schieberegister spielt in der Kryptologie eine wichtige Rolle (siehe Schneier 1991, S. 373 ff).

Another LFSR with period length 127:

```
#91:  DIM(periode?(sr(1110001, 1010101, 200))) = 127
```

Note:
It might make sense to let students produce a number of elements of a LFSR-generated sequence by hands first to understand the process – and then check it using the program, Josef

For example: start with 1,0,1,0,1,1 and use the tap sequence 1,1,0,0,1,1

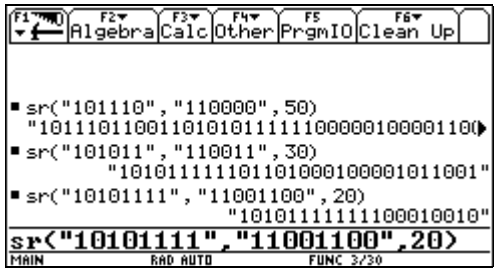


Compare with the Derive output (you don´t see the quotes in the Algebra Window!):

```
sr("101011", "110011", 12) = "101011111101"
```

Let's first investigate its periodic behaviour and then perform the calculation on the CAS-TI:

```
periode?(sr(101011, 110011, 100))

10101111111011010001000010110010101001001111000001101110011000 11

DIM(periode?(sr(101011, 110011, 100))) = 63
```



Die Schornfolge / The Schorn Sequence

In DN# 49 (S. 39) wurde Engels Aufgabe von Richard Schorn wie folgt abgeändert:
In DNL#49 (p 39) Richard Schorn varied Engel´s problem as follows:

> Eine Folge von Zahlen soll dadurch erzeugt werden, dass Einer- und Zehnerziffer (modulo 10) addiert und vorne angehängt werden, die Einerziffer aber entfällt.
>
> Generate a sequence of numbers by adding the last two digits of an integer (modulo 10) and putting the result at the beginning of the number, omit the last digit.
>
> Beispiel / Example: 3718, 9371, 8937, 0893, 2089, ...

Bei Engel geht es somit um eine Folge von Ziffern, bei Schorn um eine Folge von Zahlen – ein beträchtlicher Unterschied. Dennoch hängen beide Folgen eng miteinander zusammen.
Engel Sequence is a sequence of digits, Schorn sequence is a sequence of numbers – quite a difference, but both kinds of sequences show a tight connection.

`fillZeros(word)` is an auxiliary function to show the leading zeros (as in 0893).

```
       fillZeros(word, n) :=
         Prog
           Loop
#111:        If DIM(word) < n
                word := APPEND("0", word)
                RETURN word

       Richard(start := "3718", pickout := [2, 1], length := 10, k, n0, sum_, rseq) :=
         Prog
           k := DIM(start)
#112:      n0 := WordToNumber(start)
           sum_ := Σ(MOD(FLOOR(n, 10^(j - 1)), 10), j, pickout)
           rseq := ITERATES(FLOOR(n, 10) + 10^(k - 1)·MOD(sum_, 10), n, n0, length)
           rseq := VECTOR(fillZeros(STRING(k_), k), k_, rseq)

#113: Richard() = [3718, 9371, 8937, 0893, 2089, 7208, 8720, 2872, 9287, 5928, 0592]

#114: DIM(Richard(3718, [2, 1], ∞)) - 1 = 1560
```

Another example: Instead of taking the last two digits we take the first two ones. Derive recognizes `6798174312` as starting number of a repeating cycle.

```
Richard(1234567890, [10, 9], ∞)
[1234567890, 3123456789, 4312345678, 7431234567, 1743123456, 8174312345, 9817431234, 7981743123, 6798174312, 3679817431,
  9367981743, 2936798174, 1293679817, 3129367981, 4312936798, 7431293679, 1743129367, 8174312936, 9817431293, 7981743129,
  6798174312]

#117: DIM(Richard(5000, [2, 1], ∞)) - 1 = 15

#118: DIM(Richard(2000, [2, 1], ∞)) - 1 = 312

#119: DIM(Richard(1000, [2, 1], ∞)) - 1 = 1560
```

Richard bestätigt also die oben festgestellte Periodizität der Folge mit 5000 als erstem Element. Die Gleichung $1 \cdot 15 + 2 \cdot 312 + 6 \cdot 1560 = 9999$ festigt die Vermutung, dass es einen Zyklus mit 15, zwei Zyklen mit 312 und sechs Zyklen mit 1560 Elementen gibt. Was sagt Arthur dazu?

Richard confirms periodicity of the sequence initialized by 5000. Equation $1\times15+2\times312+6\times1560=9999$ consolidates the conjecture that there is one cycle with 15, two cycles with 312 and six cycles with 1560 elements. What is Arthur´s opinion?

```
Arthur("2000", 322) =
"2000200220242266482020222244468804684042446680248262088286004604064460806886464000400440484422864040444488826608268084882260486424066462008208028820602662828000800880868844628080888866642206426068664420862848022824006406046640204224646000600660626688246060666622284402842026228840624686044648002802082280408448282000200220"

DIM(periode?(Arthur("2000", 322))) = 312
```

Aufgaben / Tasks

1. Welcher Zusammenhang besteht zwischen den Engel- und den Schornfolgen? Kann man insbesondere aus der Periodizität der Schornfolgen auf die der Engelfolgen schließen?
   Which connection is between Engel sequences and Schorn sequences? Is it possible to deduce periodicity of Engel sequences from periodicity of Schorn sequences?
2. Untersuchen Sie die vom Schieberegister mit der Anfangsbelegung 10101110011 und dem Anzapfmuster 10100000000 erzeugte Folge. Hat sie maximale Periodenlänge?
   Investigate the sequence generated by a shift register with initial sequence 10101110011 and tap sequence 10100000000. Is this sequence of maximum period length?
3. In der Folge 1983113835952... ist jede Ziffer ab der fünften die mod-10-Summe der vier vorangehenden Ziffern. Enthält die Folge das Wort (a) 1234, (b) 3269, (c) 5198, (d) nochmals 1983? Experimentieren Sie mit verschiedenen Anfangswerten. Wie hängt die Periodenlänge vom Anfangswert ab?
   In the sequence 1983113835952... each digit starting with the fifth is the mod 10 sum of its four predecessors. Das this sequence contain the words (a) 1234, (b) 3269, (c) 3269, (d) again 1983? Experiment with various initial values. How does period legth depend on the intial value?
4. Angenommen, ein Kryptanalytiker hat die Teilfolge 00100110 eines linearen Vier-Bit-Schieberegisters abgefangen und möchte den weiteren Verlauf der Folge voraussagen. Zu diesem Zweck sucht er das Anzapfmuster zu ermitteln. Helfen Sie ihm!
   Assumed a cryptoanalyst intercepted a partial sequence 00100110 of a linear four-bit-shift register and wants to predict the following elements. For this purpose he tries to detect the tap sequence. Can you help him?

Literatur / References

Böhm, J.: Programmieren in Derive. Hagenberg: Kutzler-Verlag, 2001

Engel, A.: Eine periodische Dezimalzahl (Aufgabe P 302). In: Praxis der Mathematik 9 (1967), S. 166

Engel, A.: Mathematisches Experimentieren mit dem PC. Stuttgart: Klett-Verlag, 1991

Schneier, B.: Applied Cryptography. New York: Wiley, 1996

Welke, St.: Detection of Periods. In: DNL 49, S. 26–28

Useful related websites

```
www.elektronik-kompendium.de/sites/dig/0210211.htm
home.ecn.ab.ca/~jsavard/crypto/co040801.htm
www.newwaveinstruments.com/resources/articles/m_sequence_linear_feedback_
    shift_register_lfsr.htm
```

Coons[1, 2]-Surfaces with *DERIVE* and CAS-TI

Josef Böhm

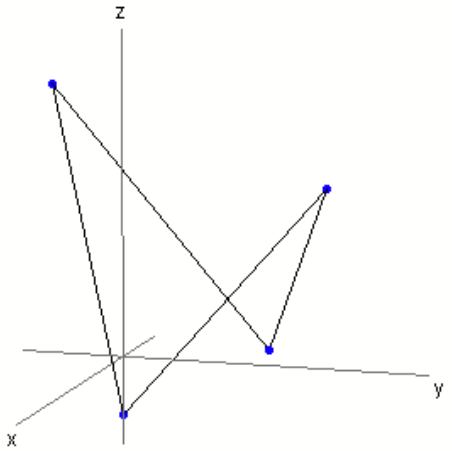
We start constructing a surface above the unit square in the first quadrant which is bounded by four connected straight lines. Two lines are lying in the *xz*-plane and *yz*-plane, the other two lines are lying in the parallel verical planes.

We have four vertices: $P_{00}(0,0,-1)$, $P_{10}(1,0,5)$, $P_{11}(1,1,1)$ and $P_{01}(0,1,3)$.
We designate the line in the *xz*-plane as *gx*0 (points on it have variable *x*-values and $y = 0$ = constant: $gx0(P_{00},P_{10})$. Analoguous definitions for $gx1(P_{01},P_{11})$, $g0y(P_{00},P_{01})$ and $g1y(P_{10},P_{11})$.

DERIVE can help producing a sketch of the situation.

(Left matrix produces the vertex points and the right one connects these points to a closed polygon in 3D space.)

$$\left[\begin{bmatrix}[0,\ 0,\ -1]\\ [1,\ 0,\ 5]\\ [1,\ 1,\ 1]\\ [0,\ 1,\ 3]\end{bmatrix},\ \begin{bmatrix}0 & 0 & -1\\ 1 & 0 & 5\\ 1 & 1 & 1\\ 0 & 1 & 3\\ 0 & 0 & -1\end{bmatrix}\right]$$



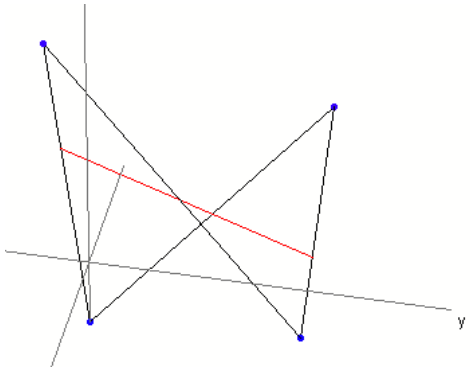
It is easy to produce the four lines in parameter form. The parameters are running from 0 to 1. We have to take care that opposite lines show the same orientation. It is no problem to name the parameters as *x* and *y*!

```
gx0(z00, z01, z10, z11, x) := [x, 0, z00 + x·(z10 − z00)]

gx1(z00, z01, z10, z11, x) := [x, 1, z01 + x·(z11 − z01)]

g0y(z00, z01, z10, z11, y) := [0, y, z00 + y·(z01 − z00)]

g1y(z00, z01, z10, z11, y) := [1, y, z10 + y·(z11 − z10)]
```

Via `[gx0(−1,3,5,1),gx1(−1,3,5,1),g0y(−1,3,5,1),g1y(−1,3,5,1)]` and `Insert > Plot` in the 3D-Plot Window we obtain the representations of the segments with boundaries 0 and 1 for the parameter. Don´t forget to activate the box `Apply parameters to rest of plot list`.

We can imagine two points X_0 and X_1 on two opposite lines *gx*0 and *gx*1 which are generated by the same parameter value *x*. We transfer these two points and the connecting segment to the graph. This segment lies parallel to the *yz*-plane and needs parameter *y* for any constant *x*.

Running *x* and *y* between 0 and 1 we receive a set of points, which form a surface in 3D-space. We found its parameter representation – with parameters *x* and *y* (which are *s* and *t* for *DERIVE*.)



```
[x1 := [x, 0, z00 + x·(z10 - z00)], x2 := [x, 1, z01 + x·(z11 - z01)]]
```

```
z(z00, z10, z11, z01) := x1 + y·(x2 - x1)
```

```
z(-1, 5, 1, 3)
```

```
[x, y, 2·x·(3 - 4·y) + 4·y - 1]
```

Inspecting the parameter form of the surface we find that the third component is nothing else than explicit representation of the *z*-values ot the surface:

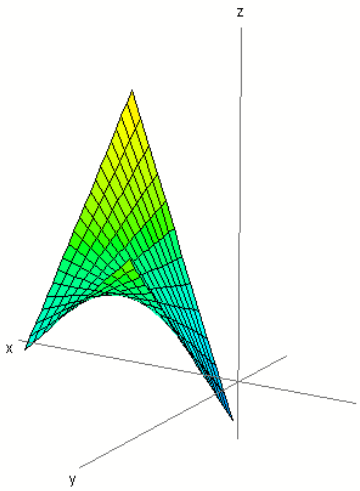
$$z(x,y) = -8xy + 6x + 4y - 1 \quad \text{mit} \quad 0 \le x, y \le 1$$

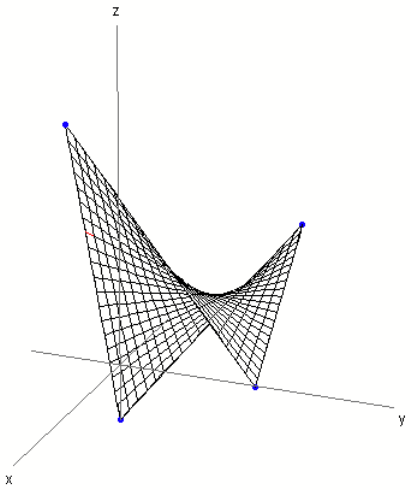
In *DERIVE* notation this reads as

```
IF(0 ≤ x ≤ 1 ∧ 0 ≤ y ≤1, 2x(3 - 4y) + 4y - 1, ?)
```

So let´s wrap this into one function depending on the *z*-values of the four corner points:

```
bilin_unitz(z00, z10, z11, z01) :=
  If 0 ≤ x ≤ 1 ∧ 0 ≤ y ≤ 1
     (z(z00, z10, z11, z01))↓3
     ?

bilin_unitz(-1, 5, 1, 3)
```





Rewriting the third component – *z*(*x*,*y*) – shows that it might be interpreted as a product of vectors and a matrix in two ways:

```
x·(y·(z00 - z01 - z10 + z11) - z00 + z10) + y·(z01 - z00) + z00
```

```
z00·x·y - z01·x·y - z10·x·y + z11·x·y - z00·x + z10·x - z00·y + z01·y + z00
```

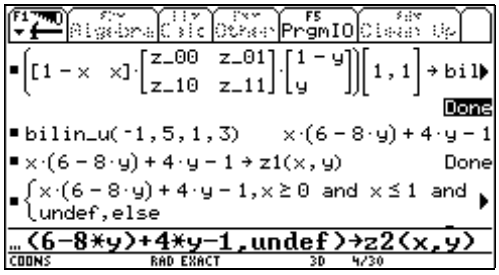
```
(1 - y)·((1 - x)·z00 + x·z10) + y·((1 - x)·z01 + x·z11)
```

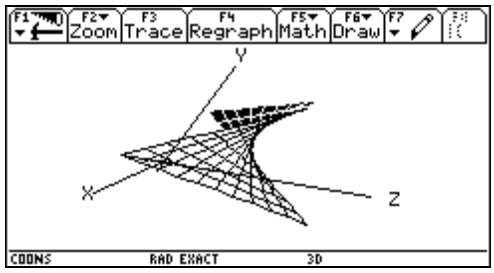
$$[1 - x,\ x]\cdot\begin{bmatrix} z00 & z01 \\ z10 & z11 \end{bmatrix}\cdot\begin{bmatrix} 1 - y \\ y \end{bmatrix}$$

```
[x·(y·(z00 - z01 - z10 + z11) - z00 + z10) + y·(z01 - z00) + z00]
```

$$[1 - y,\ y]\cdot\begin{bmatrix} z00 & z10 \\ z01 & z11 \end{bmatrix}\cdot\begin{bmatrix} 1 - x \\ x \end{bmatrix}$$

```
[x·(y·(z00 - z01 - z10 + z11) - z00 + z10) + y·(z01 - z00) + z00]
```





This leads to another – and more compact - definition of the surface:

```
bilin_unit(z00, z10, z11, z01) :=
  If 0 ≤ x ≤ 1 ∧ 0 ≤ y ≤ 1
     ([1 - x, x]·[z00, z01; z10, z11]·[1 - y; y])↓1
     ?

bilin_unit(-1, 5, 1, 3)
```

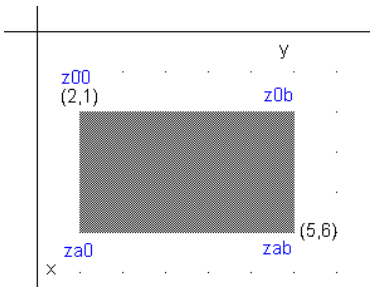
First generalization

The base figure shall no longer be only a square but a rectangle with $x0 \le x \le x1$ and $y0 \le y \le y1$. We change the vertices to

$$P_{00}(0,0,1),\ P_{50}(5,0,6),\ P_{53}(5,3,-1) \text{ and } P_{03}(0,3,2).$$

Parameter representation of the segments must be changed for accepting the default parameter boundaries 0 and 1 to run from $x0$ to $x1$ and from $y0$ to $y1$.

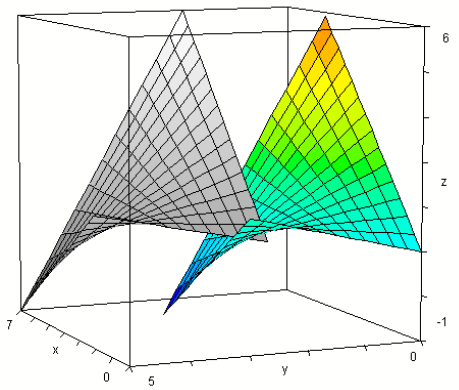
The sketch shows a top view for $(x0,y0) = (2,1)$ and $(x1,y1) = (5,6)$. Default values for function `bilin_rect()` are (0,0) and (1,1).




```
bilin_rect(z00, za0, zab, z0b, x0:=0, y0:=0, x1:=1, y1:=1, zxy):=
         prog(
               zxy:=([1 - (x - x0)/(x1 - x0), (x - x0)/(x1 - x0)]·
                     [z00, z0b; za0, zab]·
                     [1 - (y - y0)/(y1 - y0); (y - y0)/(y1 - y0)])↓1,
              IF(x0 ≤ x ≤ x1 ∧ y0 ≤ y ≤ y1, zxy, ?)
             )

bilin_rect(-1, 5, 1, 3) = IF(x ≤ 1 ∧ y ≤ 1 ∧ x ≥ 0 ∧ y ≥ 0, 2·x·(3 - 4·y) + 4·y - 1, ?)
```

This function seems to work properly and we try to produce two parallel surfaces.

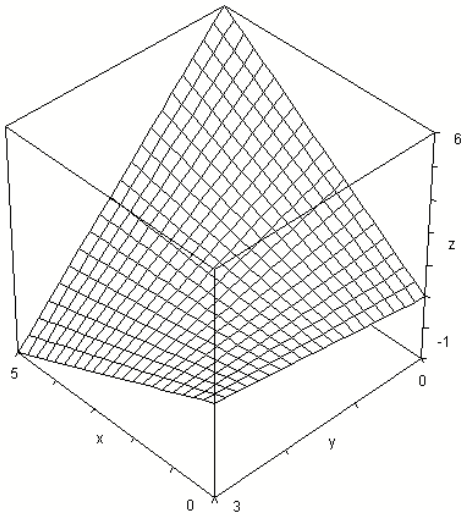
```
bilin_rect(1, 6, -1, 2, 0, 0, 5, 3)

IF[x ≤ 5 ∧ y ≤ 3 ∧ x ≥ 0 ∧ y ≥ 0, (5·(y + 3) - x·(8·y - 15))/15, ?]

bilin_rect(1, 6, -1, 2, 2, 2, 7, 5)

IF[x ≤ 7 ∧ y ≤ 5 ∧ x ≥ 2 ∧ y ≥ 2, (3·(7·y - 19) - x·(8·y - 31))/15, ?]
```

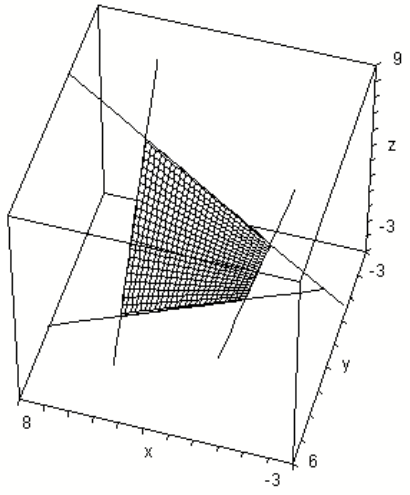


Additional exercise:

Find the boundary lines and scale in such a way that the range for the parameter is from 0 to 1 for all segments. (see *DERIVE* file).

```
[zrx0(1, 2, 6, -1, 0, 0, 5, 3), zrxb(1, 2, 6, -1, 0, 0, 5, 3),
 zry0(1, 2, 6, -1, 0, 0, 5, 3), zrya(1, 2, 6, -1, 0, 0, 5, 3)]
```





It might be useful for students to repeat the relationship between the parameter form and explicit form of the surface:

```
[z_00 := [0, 0, 1], z_0b := [0, 3, 2], z_ab := [5, 3, -1], z_a0 := [5, 0, 6]]

[g_ := z_a0 + t·(z_00 - z_a0), h_ := z_ab + t·(z_0b - z_ab)]

g_ + s·(h_ - g_)
```

$$[5 - 5 \cdot t,\ 3 \cdot s,\ s \cdot (8 \cdot t - 7) - 5 \cdot t + 6]$$

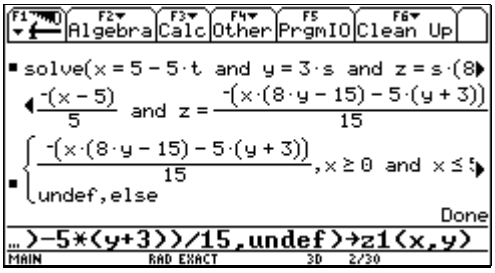
```
SOLVE(x = 5 - 5·t ∧ y = 3·s ∧ z__ = s·(8·t - 7) - 5·t + 6, [s, t, z__])
```

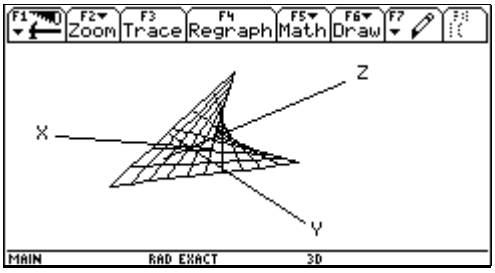
$$s = \frac{y}{3} \wedge t = \frac{5 - x}{5} \wedge z\_\_ = \frac{5 \cdot (y + 3) - x \cdot (8 \cdot y - 15)}{15}$$

$$\frac{5 \cdot (y + 3) - x \cdot (8 \cdot y - 15)}{15}$$

$$\text{IF}\left[x \le 5 \wedge y \le 3 \wedge x \ge 0 \wedge y \ge 0,\ \frac{5 \cdot (y + 3) - x \cdot (8 \cdot y - 15)}{15},\ ?\right]$$

It is not really difficult to transfer this to the TI-89/TI-92/Voyage 200:





Second generalization

The second generalization is much more challenging: two curves lying in opposite parallel vertical planes form the boundary of the surface. We denote the pair of curves either as ($zx0(x)$, $zxb(x)$) – lying in planes perpendicular to the y-axis or as ($z0y(y)$,$zay(y)$) if lying in two planes perpendicular to the x-axis.

Let´s choose the following base rectangle [(1,2), (5,2,), (5,10), (1,10)].

As first example I´ll take two parabolas, which should form a surface for $1 \le x \le 5$ and $2 \le y \le 10$.

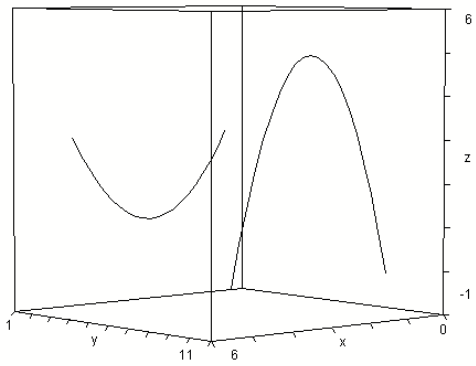
$$\left[zx0 := \frac{x^2 - 6\cdot x + 11}{2},\ zxb := -\frac{5\cdot(x^2 - 6\cdot x + 5)}{4}\right]$$

For representing the two boundary curves in R^3 we have to create space curves. This shall be automated: *DERIVE* shall learn from the appearing variables – x or y – which type is needed.

```
c0(kurve, x0 := 0, y0 := 0, x1 := 1, y1 := 1) :=
  If (VARIABLES(kurve))↓1 = x
     SUBST([x, y0, kurve], x, x0 + t·(x1 - x0))
     SUBST([x0, y, kurve], t, y0 + t·(y1 - y0))
     SUBST([x0, y, kurve], y, y0 + t·(y1 - y0))

c1(kurve, x0 := 0, y0 := 0, x1 := 1, y1 := 1) :=
  If (VARIABLES(kurve))↓1 = x
     SUBST([x, y1, kurve], x, x0 + t·(x1 - x0))
     SUBST([x1, y, kurve], y, y0 + t·(y1 - y0))
     SUBST([x1, y, kurve], y, y0 + t·(y1 - y0))
```

$$c0(zx0,\ 1,\ 2,\ 5,\ 10) = \left[4\cdot t + 1,\ 2,\ 8\cdot t^2 - 8\cdot t + 3\right]$$

$$c1(zxb,\ 1,\ 2,\ 5,\ 10) = [4\cdot t + 1,\ 10,\ 20\cdot t\cdot(1 - t)]$$

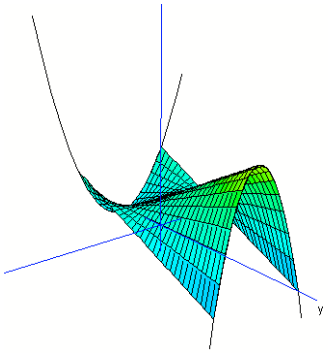


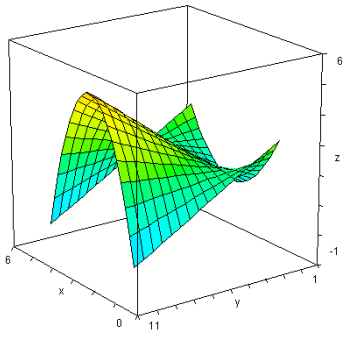
Remaining work is not difficult. Pairs of points of the two "profile curves" $c0$ and $c1$, which are generated by the same parameter value must be connected.

$\left((1-\frac{y}{b})\cdot c0+\frac{y}{b}\cdot c1\right)$ oder $\left(1-\frac{x}{a}\right)\cdot c0+\frac{x}{a}\cdot c1$ is the "raw form", assuming one cornerpoint of the base rectangle being the origin with legths a and b of the rectangle sides. Let´s define a general function and we don´t have to take care for scaling in the future:

```
bi_curves(k0,k1,x0:=0,y0:=0,x1:=1,y1:=1,zxy):=prog(
              zxy:=IF((VARIABLES(k0))↓1=x,
                        (1-(y-y0)/(y1-y0))·k0+(y-y0)/(y1-y0)·k1,
                        (1-(x-x0)/(x1-x0))·k0+(x-x0)/(x1-x0)·k1,
                        (1-(x-x0)/(x1-x0))·k0+(x-x0)/(x1-x0)·k1),
                   IF(x0 ≤ x ≤ x1 ∧ y0 ≤ y ≤ y1, zxy, ?)
                                                  )
```

```
bi_curves(zx0,zxb,1,2,5,10)
```

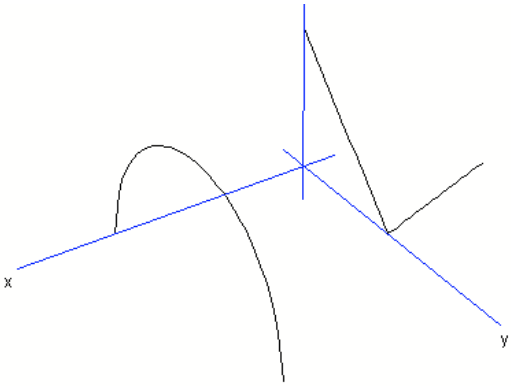


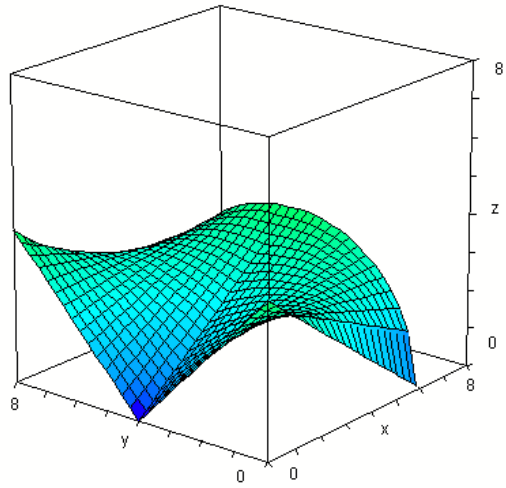


Next example shows boundary curves lying in planes parallel to *yz*-plane. The surface is defined over the region described by $0 \le x \le 6$ and $0 \le y \le 8$.

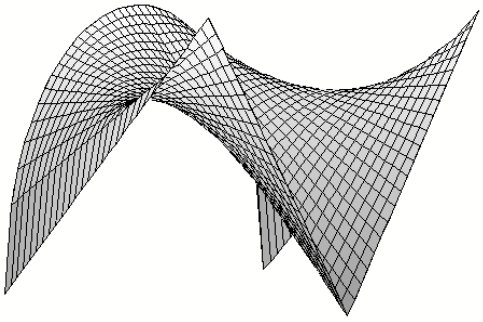
$$\left[c0(|y-4|,\ 0,\ 0,\ 6,\ 8),\ c1\left(\sqrt{(16-(y-4)^2)},\ 0,\ 0,\ 6,\ 8\right)\right]$$

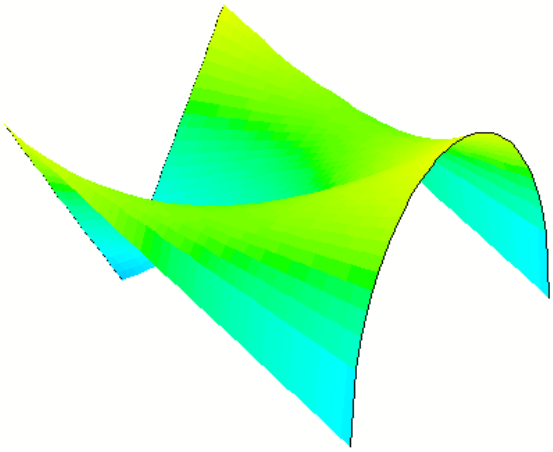
$$bi\_curves(|y-4|,\ \sqrt{(16-(y-4)^2)},\ 0,\ 0,\ 6,\ 8)$$



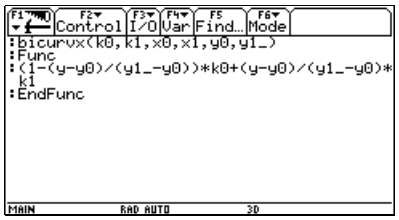


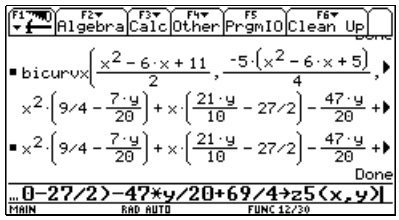
In *DERIVE* 6 we can suppress plotting the grid lines which leads to a kind of photo realistic pictures.

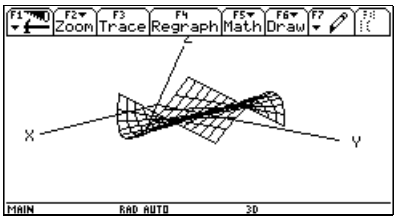


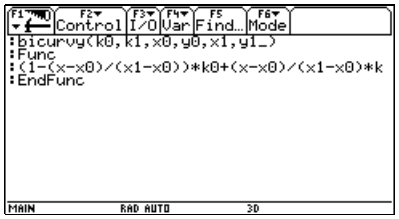


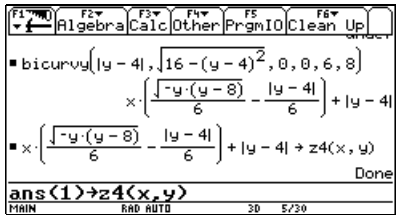
And how does the TI-calculator perform?

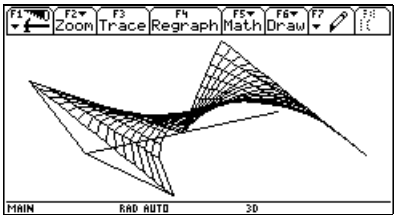




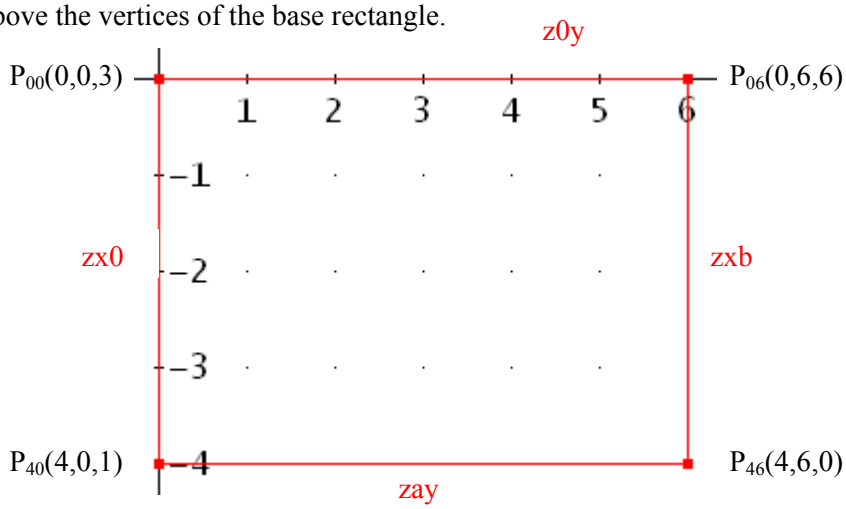








The last generalization is the most difficult one and it finally leads to the Coons-Surfaces. All boundary lines are - nonlinear – curves. Base figure is again a rectangle and the boundary curves are connected above the vertices of the base rectangle.



We produce four parabolas using quadratic regression.

par1: in xz-plane (y = 0) passing (4,0,1), (0,0,3), (2,0,4)
par2: in der plane y = 6 passing (4,6,0), (0,6,6), (3,6,-2)
$0 \le x \le 4$

#58: $\text{FIT}\left(\left[x,\ a\cdot x^2+b\cdot x+c\right],\ \begin{bmatrix}4&1\\0&3\\2&4\end{bmatrix}\right) = -\frac{x^2}{2}+\frac{3\cdot x}{2}+3$

#59: $\text{FIT}\left(\left[x,\ a\cdot x^2+b\cdot x+c\right],\ \begin{bmatrix}4&0\\0&6\\3&-2\end{bmatrix}\right) = \frac{7\cdot x^2}{6}-\frac{37\cdot x}{6}+6$

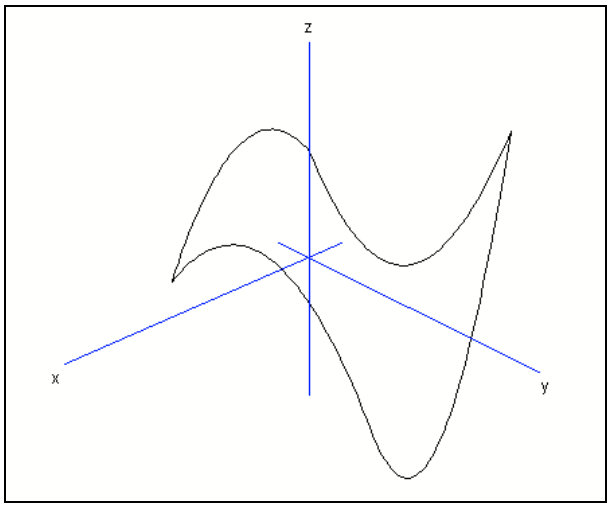
par3: in plane x=0 passing (0,0,3), (0,2,1), (0,6,6)
par4: in plane x = 4 passing (4,0,1), (4,3,3), (4,6,0)
$0 \le y \le 6$

#60: $\text{FIT}\left(\left[y,\ a\cdot y^2+b\cdot y+c\right],\ \begin{bmatrix}0&3\\6&6\\2&1\end{bmatrix}\right) = \frac{3\cdot y^2}{8}-\frac{7\cdot y}{4}+3$

#61: $\text{FIT}\left(\left[y,\ a\cdot y^2+b\cdot y+c\right],\ \begin{bmatrix}0&1\\3&3\\6&0\end{bmatrix}\right) = -\frac{5\cdot y^2}{18}+\frac{3\cdot y}{2}+1$

Ans we plot these four curves:

$$\left[c0\left[-\frac{x^2}{2}+\frac{3\cdot x}{2}+3,\ 0,\ 0,\ 4,\ 6\right],\ c1\left[\frac{7\cdot x^2}{6}-\frac{37\cdot x}{6}+6,\ 0,\ 0,\ 4,\ 6\right],\ c0\left[\frac{3\cdot y^2}{8}-\frac{7\cdot y}{4}+3,\ 0,\ 0,\ 4,\ 6\right],\ c1\left[-\frac{5\cdot y^2}{18}+\frac{3\cdot y}{2}+1,\ 0,\ 0,\ 4,\ 6\right]\right]$$



In our first approach we will assume that one vertex of the base rectangle is the origin. Legths of the sides of the rectangle are a (in x-direction) and b (in y-direction). This makes presentation clearer. In the original papers only the unitsquare ($a = 1$, $b = 1$) is used.

$$\left[zx0 := -\frac{x^2}{2} + \frac{3\cdot x}{2} + 3,\ zxb := \frac{7\cdot x^2}{6} - \frac{37\cdot x}{6} + 6,\ z0y := \frac{3\cdot y^2}{8} - \frac{7\cdot y}{4} + 3,\ zay := -\frac{5\cdot y^2}{18} + \frac{3\cdot y}{2} + 1\right]$$

$$[a := 4,\ b := 6]$$

$$[z00 := 3,\ z0b := 6,\ zab := 0,\ za0 := 1]$$

The third expression contains the z-values of the "corner points", where the boundary curves connect. Later we will automate this, too.

What follows now is the important and surprising point – for students and for teachers as well.

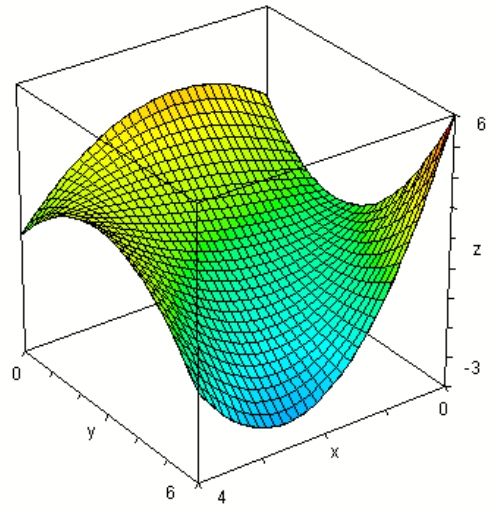
The following matrix product delivers the explicit form of the surface:

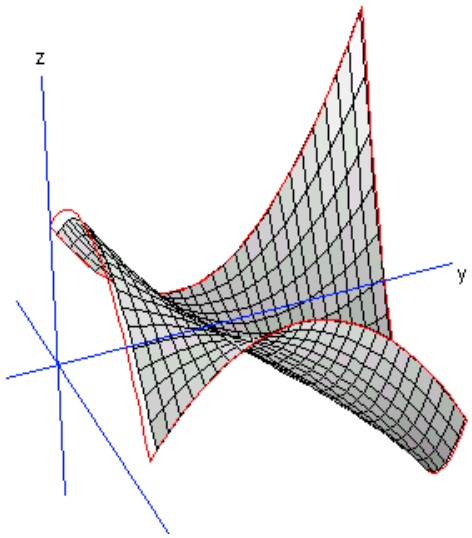
$$\left[1 - \frac{x}{a},\ \frac{x}{a},\ 1\right]\cdot\begin{bmatrix} -z00 & -z0b & z0y \\ -za0 & -zab & zay \\ zx0 & zxb & 0 \end{bmatrix}\cdot\begin{bmatrix} 1 - \frac{y}{b} \\ \frac{y}{b} \\ 1 \end{bmatrix}$$

$$\left[\frac{16\cdot x^2\cdot(5\cdot y - 9) - x\cdot(47\cdot y^2 + 86\cdot y - 432) + 36\cdot(3\cdot y^2 - 14\cdot y + 24)}{288}\right]$$

$$\text{IF}\left[0 \le x \le a \wedge 0 \le y \le b,\ \frac{16\cdot x^2\cdot(5\cdot y - 9) - x\cdot(47\cdot y^2 + 86\cdot y - 432) + 36\cdot(3\cdot y^2 - 14\cdot y + 24)}{288},\ ?\right]$$

So we can proceed plotting the surface:





Before defining a working function we will try to derive the "Coons-formula". For this purpose we need an auxiliary function $u(x,y)$ which decribes the boundary curves. (This part follows the references, but we will use *DERIVE* for the calculations.)

First of all we add the two different forms of respresentation from `bicurves()`.

$$z(x,y)=\left(1-\frac{y}{b}\right)\cdot ux0+\frac{y}{b}\cdot uxb+\left(1-\frac{x}{a}\right)\cdot u0y+\frac{x}{a}\cdot uay\ .$$

It is for sure that this surface does not contain the boundary curves!

DERIVE enters the stage.

We take care that *ux*0 and *uxb* are lying on the surface.

$$u(x,\ y):=$$

$$z(x,\ y):=\left(1-\frac{y}{b}\right)\cdot u(x,\ 0)+\frac{y}{b}\cdot u(x,\ b)+\left(1-\frac{x}{a}\right)\cdot u(0,\ y)+\frac{x}{a}\cdot u(a,\ y)$$

$$z(x,\ 0)=u(x,\ 0)+\frac{x\cdot u(a,\ 0)}{a}+\frac{(a-x)\cdot u(0,\ 0)}{a}$$

As z(x,0) and u(x,0) must coincide we have to subtract the two superfluous expressions from z(x,y)

$$z(x,\ y):=\left(1-\frac{y}{b}\right)\cdot\left(u(x,\ 0)-\frac{x\cdot u(a,\ 0)}{a}-\frac{(a-x)\cdot u(0,\ 0)}{a}\right)+\frac{y}{b}\cdot u(x,\ b)+\left(1-\frac{x}{a}\right)\cdot u(0,\ y)+\frac{x}{a}\cdot u(a,\ y)$$

And z(x,b) must coincide with u(x,b), therefore

$$z(x,\ b)=u(x,\ b)+\frac{x\cdot u(a,\ b)}{a}+\frac{(a-x)\cdot u(0,\ b)}{a}$$

$$z(x,\ y):=\left(1-\frac{y}{b}\right)\cdot\left(u(x,\ 0)-\frac{x\cdot u(a,\ 0)}{a}-\frac{(a-x)\cdot u(0,\ 0)}{a}\right)+\frac{y}{b}\cdot\left(u(x,\ b)-\frac{x\cdot u(a,\ b)}{a}-\frac{(a-x)\cdot u(0,\ b)}{a}\right)+\left(1-\frac{x}{a}\right)\cdot u(0,\ y)+\frac{x}{a}\cdot u(a,\ y)$$

Let´s have a check:

#77: [z(x, 0), z(x, b)] = [u(x, 0), u(x, b)]

This does fit!!!

and what is much more surprising:

#78: [z(0, y), z(a, y)] = [u(0, y), u(a, y)]

The other pair of curves are also part of the surface now!

I rewrite right side of the surface definition using z00, zxb, zay, zab etc.

$$\left(1-\frac{y}{b}\right)\cdot\left(zx0-\frac{x\cdot za0}{a}-\frac{(a-x)\cdot z00}{a}\right)+\frac{y}{b}\cdot\left(zxb-\frac{x\cdot zab}{a}-\frac{(a-x)\cdot z0b}{a}\right)+\left(1-\frac{x}{a}\right)\cdot z0y+\frac{x}{a}\cdot zay$$

$$test := \frac{a\cdot(y\cdot(z00-z0b-zx0+zxb)-b\cdot(z00-z0y-zx0))-x\cdot(y\cdot(z00-z0b-za0+zab)-b\cdot(z00-z0y-za0+zay))}{a\cdot b}$$

and have the question if this expression is identical with the matrix product from above?

$$\#81:\quad test-\left(\begin{bmatrix}1-\frac{x}{a} & \frac{x}{a} & 1\end{bmatrix}\cdot\begin{bmatrix}-z00 & -z0b & z0y\\ -za0 & -zab & zay\\ zx0 & zxb & 0\end{bmatrix}\cdot\begin{bmatrix}1-\frac{y}{b}\\ \frac{y}{b}\\ 1\end{bmatrix}\right)_1=0$$

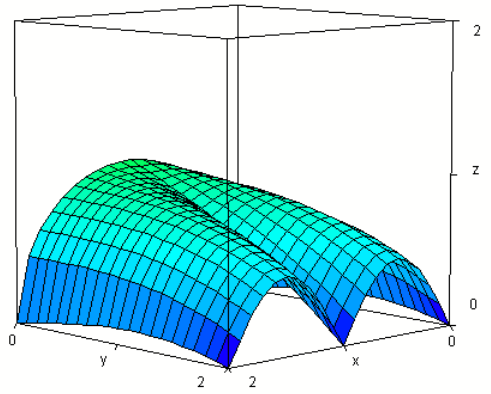
YES!!

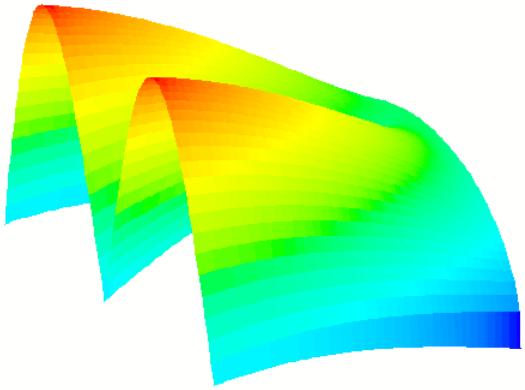
So we can finish, defining an appropriate function for the general case:

```
coons(zx0,zay,zxb,z0y,x0:=0,y0:=0,x1:=1,y1:=1,z00,z0b,za0,zab,zxy):=
          PROG(
               "@ die z-values in the corner points",
               z00 := LIM(zx0, x, x0),
               z0b := LIM(z0y, y, y1),
               za0 := LIM(zx0, x, x1),
               zab := LIM(zay, y, y1),
               zxy := ([1 - (x - x0)/(x1 - x0), (x - x0)/(x1 - x0), 1]·
                       [-z00, -z0b, z0y; -za0, -zab, zay; zx0, zxb, 0]·
                       [1 - (y - y0)/(y1 - y0); (y - y0)/(y1 - y0); 1])↓1,
               IF(x0 ≤ x ≤ x1 ∧ y0 ≤ y ≤ y1, zxy, î)
              )
```

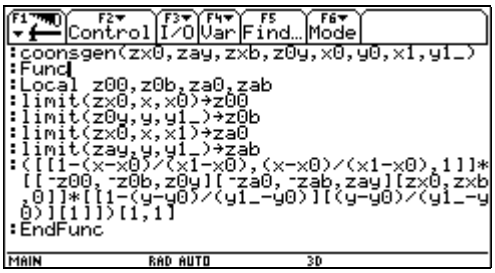
I´ll show some examples to inspire your own experiments.

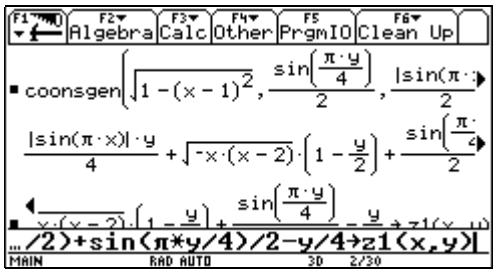
$$coons\left(\sqrt{(1-(x-1)^2)},\ \frac{SIN\left(\frac{\pi\cdot y}{4}\right)}{2},\ \frac{|SIN(\pi\cdot x)|}{2},\ \frac{SIN\left(\frac{\pi\cdot y}{4}\right)}{2},\ 0,\ 0,\ 2,\ 2\right)$$

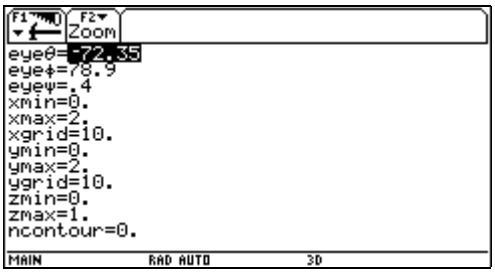


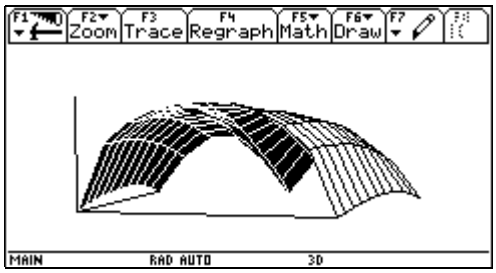


As you can see it works also on the TI-92 / Voyage 200



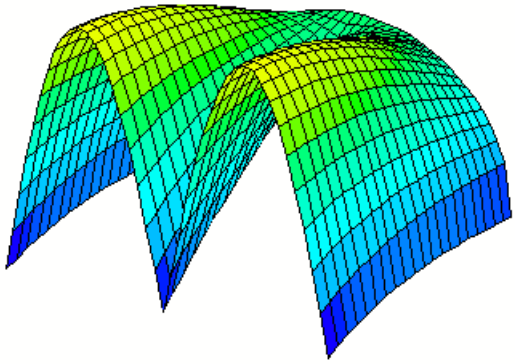






A variation of the surface from above:

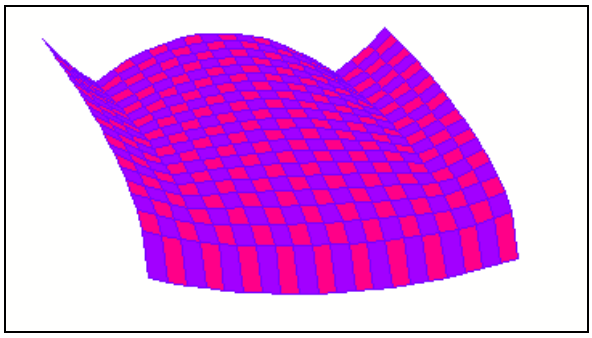
$$\text{coons}\left(\sqrt{(1-(x-1)^2)},\ \frac{\text{SIN}\left(\frac{\pi\cdot y}{4}\right)}{2},\ \frac{3\cdot|\text{SIN}(\pi\cdot x)|}{2},\ \frac{\text{SIN}\left(\frac{\pi\cdot y}{4}\right)}{2},\ 0,\ 0,\ 2,\ 2\right)$$

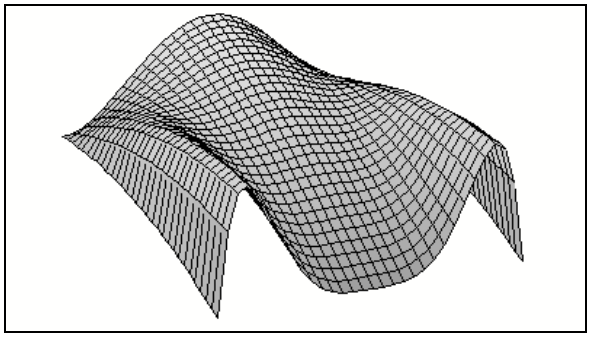


The base of the next two surfaces is the unit square $0 \le x \le 1$ and $0 \le y \le 1$.

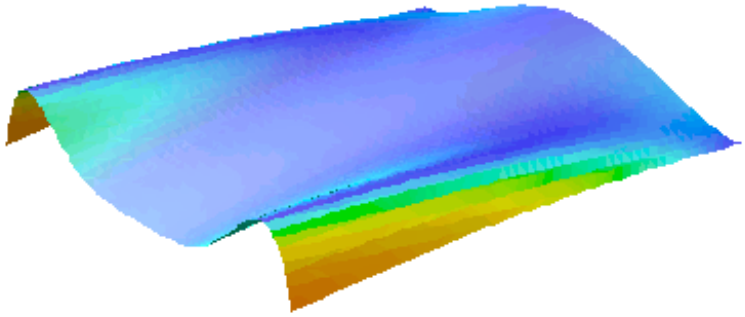
#85: $\text{coons}\left(\frac{\left|8\cdot x^2 - 8\cdot x + 1\right| + 5}{8}, \frac{2\cdot\sqrt{(1-y)} + 1}{4}, \frac{x^2 - x + 1}{4}, \frac{2\cdot\sqrt{(1-y)} + 1}{4}\right)$

#86: $\text{coons}\left(2\cdot x^2\cdot(1-x)^2 + \frac{1}{5}, \frac{2-y^2}{10}, -4\cdot x\cdot(x-1)\cdot(2\cdot x-1)^4 + \frac{1}{10}, \frac{2-y^2}{10}\right)$





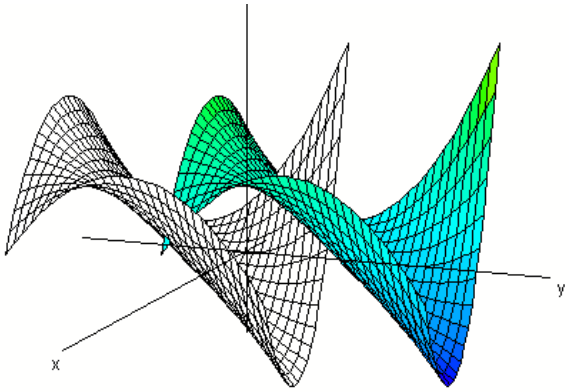
It is not difficult to transfer the surface formula to *DPGraph:*

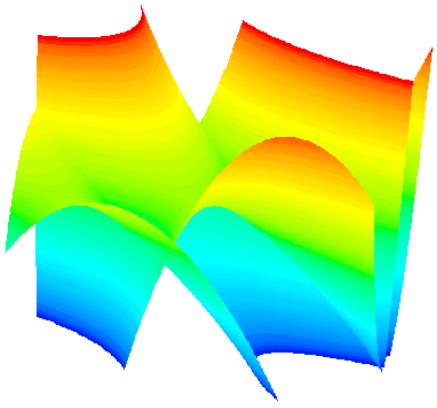


We plot again the surface from above bounded by four parabolas together with a second surface which results of a translation of the first one in order to test function coons().

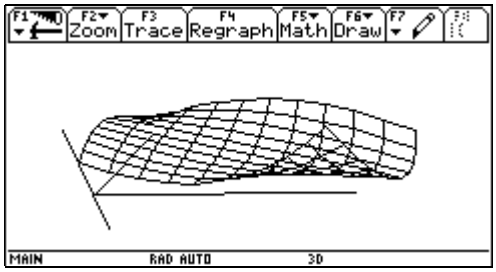
#87: $f11(x, y) := \text{coons}\left(-\frac{x^2}{2} + \frac{3\cdot x}{2} + 3, -\frac{5\cdot y^2}{18} + \frac{3\cdot y}{2} + 1, \frac{7\cdot x^2}{6} - \frac{37\cdot x}{6} + 6, \frac{3\cdot y^2}{8} - \frac{7\cdot y}{4} + 3, 0, 0, 4, 6\right)$

#88: $f12(x, y) := \text{coons}\left(-\frac{(x-1)^2}{2} + \frac{3\cdot(x-1)}{2} + 3, -\frac{5\cdot(y+3)^2}{18} + \frac{3\cdot(y+3)}{2} + 1, \frac{7\cdot(x-1)^2}{6} - \frac{37\cdot(x-1)}{6} + 6, \frac{3\cdot(y+3)^2}{8} - \frac{7\cdot(y+3)}{4} + 3, 1, -3, 5, 3\right)$





The TI-presentation looks a bit poor after the great *DERIVE* and *DPGraph* plots, but I find it remarkable what can be done with these small devices algebraically and graphically as well.



Finally I´d like to check my function once more comparing the function values of the two surfaces `fl1` and `fl2`.

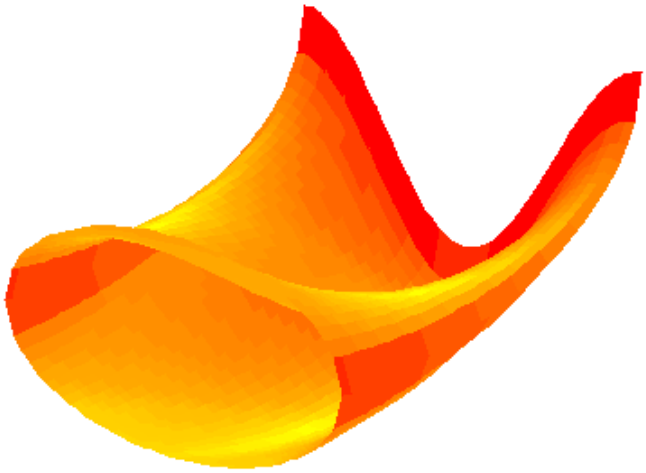
```
values1 := VECTOR(VECTOR(fl1(x, y), y, 0, 6), x, 0, 4)

values2 := VECTOR(VECTOR(fl2(x, y), y, -3, 3), x, 1, 5)
```

$$\text{values1} - \text{values2} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

So we don´t have to rely only on the optical impression but have an additional numerical confirmation.

Final comment

In April I had the occasion to give a workshop on Coons-Surfaces.
The colleagues – teachers from Tyrol – enjoyed experimenting and creating their own surfaces. So they designed their fancy cars, etc.



My friend Heiner Juen composed two surfaces for "Josef´s Helmet". Many thanks Heiner. I´ll keep it in honour.

References:

I found the Coons-Surfaces in a Belgian Teachers´ Journal [2]. Jan Vermeylen helped providing part 1 [1] of this contribution which contains the basics. Many thanks for this together with best regards to Belgium.

[1] H.van Looy, *Coonsoppervlakken bij computer aided design*, Wiskunde & Onderwijs, 1990

[2] L. Van den Broeck, *Motorkappen en Coonsoppervlakken*, Wiskunde & Onderwijs, 1996

This is a related website::
`http://olli.informatik.uni-oldenburg.de/Grafiti3/grafiti/flow11/page3.html`
See also additional webpages provided by Ernest Carpenter (User Forum).

Introduction to Global Position System

Two Dimensional Preliminaries

Carl Leinbach, Bigglerville, USA, leinbach@gettysburg.edu

NOTE: While working with this lesson, we recommend that you open a 2D-Plot Window and then choose the option Window > Tile > Vertically. Later in this lesson, you will be asked to modify the range of the window using the Plot Window options Set > Plot Range. Using this screen format, you will be able to follow the lesson and view the corresponding graphics displays that you will generate.

Acknowledgement: This activity was inspired by a talk given by Mr. Ralph Irons of Central Shenandoah Valley Regional Governor's School at the 2001 T^3 International Conference in Colombus, OH. His presentation can be found in the Proceedings of that Confence call "2001 a Technological Odyssey." The presentation given here is distinctly different from that of Mr. Irons, but his excitement and exposition were the inspiration.

Background - Step 1:

The classical definition of a circle is: *the locus of all points equidistant from a fixed point.* The measure of the distance from the fixed point is called the *radius* of the circle and the fixed point itself is called the *center* of the circle. In the cartesian coordinate system a point is designated by a pair of real numbers, say (c1, c2). Thus, we need three parameters to completely describe a circle: c1, the *x*-coordinate of the center; c2, the *y*-coordinate of the center; and r, the radius of the circle. Analytically, this tells us that the definition of a circle is the set of all points, (x, y) that satisfy the following equation based on the distance formula:

$$\#1: \quad (x - c1)^2 + (y - c2)^2 = r^2$$

Derive allows you to plot expressions such as #1 provided the values of the parameters are defined.

For example, if you wished to see the graph of a circle with center at (-1, 2) and radius 3, you simply highlight expression #1, click on the SUB toolbar button and type in the respective new values for c1, c2, and r variables (parameters). Click on OK to replace the parameters with your values. Switch to the plot window and click on the plot toolbar button or click Insert > Plot. If you receive a message that says "Unable to Plot Highlighted Expression", open a 2D-plot window and click on Window > Approximate Before Plotting and try again.

The resulting plot is the set of points, (x, y), satisfying the new values substituted in expression #1.

What if you know some points on a circle and want to find the equation of the circle? This is the reverse of our previous example. We need to find the values of the parameters c1, c2, and r. Once again, expression #1 is our road to success. The first question we need to address is, how many points uniquely determine a circle? Since we have three parameters to determine, it is sensible to guess that three points should do the job. Of course, the points can not be co-linear. Let's give it a try.

Suppose we wish to find a circle passing through the three points (1, 1), (-2, 0), and (-1. -1).

First we copy expression #1 into the Derive Command Entry Line by highlighting it and pressing the F3 key. Substitute 1 for x and 1 for y. Press the Enter key. The expression will remain in the Command Entry Line. Alter the expression to reflect entering -2 for x and 0 for y. Press enter again. Repeat the process for the point (-1, -1).

This results in the following three equations.

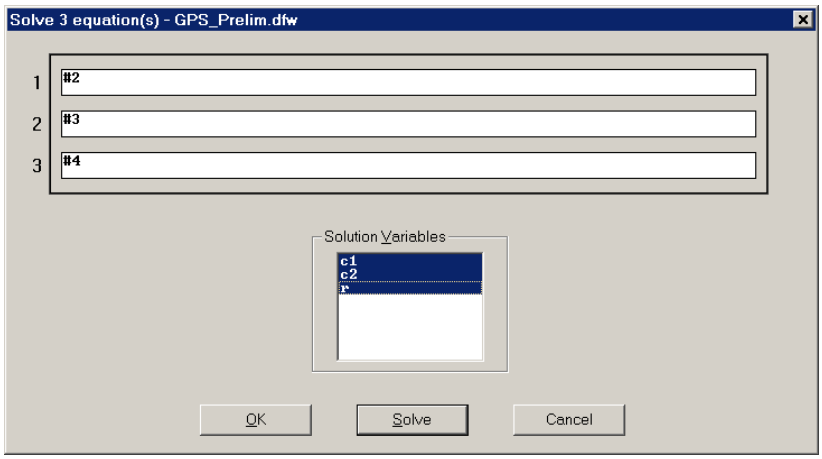
$$\#2: \quad (1 - c1)^2 + (1 - c2)^2 = r^2$$

$$\#3: \quad (-2 - c1)^2 + (0 - c2)^2 = r^2$$

$$\#4: \quad (-1 - c1)^2 + (-1 - c2)^2 = r^2$$

Derive has a very powerful solver for systems of polynomial equations.

Press Solve > System. An inquiry box will ask you how many equations you wish to solve. Indicate 3. A display similar to the one shown below appears. Since you have the equations as part of your Derive worksheet, you need only enter the numbers of the expressions that have the three equations as shown below. Click on the blank space in the Solution Variables area and then click Solve.



The following two expressions are displayed on your Derive worksheet. Note that in expression #6 there are two answers. Why?

$$\#5: \quad \text{SOLVE}\left[(1 - c1)^2 + (1 - c2)^2 = r^2,\ (-2 - c1)^2 + (0 - c2)^2 = r^2,\ (-1 - c1)^2 + (-1 - c2)^2 = r^2\right], [c1, c2, r])$$

$$\#6: \quad \left[c1 = -\frac{1}{2} \wedge c2 = \frac{1}{2} \wedge r = \frac{\sqrt{10}}{2},\ c1 = -\frac{1}{2} \wedge c2 = \frac{1}{2} \wedge r = -\frac{\sqrt{10}}{2}\right]$$

At this point we enter the solution for c1, c2, and r into expression #1 and plot the circle.

$$\#7: \quad \left(x - -\frac{1}{2}\right)^2 + \left(y - \frac{1}{2}\right)^2 = \left(\frac{\sqrt{10}}{2}\right)^2$$

Exercise

1. Choose three arbitrary non colinear points on the plane and determine the equation of a circle through these three points. Graph the resulting circle.

Background - Step 2

Now that we have done a brief study of circles, we will consider how they may be applied to GPS System. Let's suppose that you are in a flat plane located at the point having grid coordinates (-1, 2), but you don't know that fact. Suppose there are transmitters located at grid coordinates, (3, 0), (-3, 1), and (0, 1). What you do know is that you are located at distances of the square root of 20, the square root of 5, and the square root of 2, respectively from these three points. You will use that fact to locate your position on the plane.

The first thing that you do is write the equations for the three circles centered at the transmitters and having radii equal to their distance from your position on the plane.

$$\#8: \quad (x - 3)^2 + (y - 0)^2 = \sqrt{20}^2$$

$$\#9: \quad (x - -3)^2 + (y - 1)^2 = \sqrt{5}^2$$

$$\#10: \quad (x - 0)^2 + (y - 1)^2 = \sqrt{2}^2$$

Once we have the equations, it is natural to look at their graphs and find the intersection point(s).

Highlight each of the expressions in the algebra window and plot them.

Note that even though there are only two independent variables, you need to plot all three of the expressions to find your position. Why?

You can solve the equations exactly using the Derive Solve > System command as we did before. The result of this command is shown below. Note that even though we had only two unknowns, we entered three equations.

$$\#11: \quad \text{SOLVE}\left[(x - 3)^2 + (y - 0)^2 = \sqrt{20}^2 \text{ , } (x - -3)^2 + (y - 1)^2 = \sqrt{5}^2 \text{ , } (x - 0)^2 + (y - 1)^2 = \sqrt{2}^2 \right], [x, y])$$

$$\#12: \quad [x = -1 \wedge y = 2]$$

At this point, let's review what you know about circles. In the following exercises draw a graph to illustrate that your answer is correct.

Exercises:

2. Is it possible to find c1, c2, and r so that expression #1 has exactly one point, (x, y) that will satisfy the equation.

3. Is it possible for two distinct circles to intersect at no point? one point?

two points?

4. Is it possible for three distinct circles to intersect at no point? one point? two points?

5. Why is it not possible to find three or more distinct circles that intersect in exactly three points?

A "2 Dimensional" GPS

NOTE: The velocity given for the signal sent from the transmitters has no relation to the actual speed with which signals are sent from GPS sattelites or radio transmitters. We are using this velocity only to keep the magnitude of the numbers we are dealing with more manageable so that the numbers do not get in the way of the exposition. In part II we will deal with a more realistic situation.

Prior to beginning this unit, resize your graphics screen so that the range on the x-axis is (-300, 900) with 4 intervals, and the range on the y-axis is (-200 , 800) with 5 intervals. Also,because accuracy is important in GPS calculations set the decimal accuracy to 10 digits. Use the Algebra Window's Declare > Simplification Settings to set the Precision Digits to 10.

You know that you are located at a point somewhere in this plane region. Assume that scale on the graphics screen is in miles. You activate your GPS to locate your position. It receives signals from three transmitters that have been placed in the plane at locations of loc1=(75, 150), loc2=(510, 110), and loc3=(150,20). The transmitters all know their own position and the exact time as determined by an atomic clock in the transmitter. The transmitter clocks are synchronized on a regular basis so that the time at each transmitter is absolutely accurate. Each transmitter broadcasts its coordinates and the time. The broadcast signal travels at 200 miles per second.

Your GPS is able to receive the signals and record the time that each signal is received. However, your GPS does not have a clock that is synchronized with the transmitters. Thus, there is a time time discrepency, say d, between the time recorded by your GPS receiver and the the actual time as determined by the synchronized atomic clocks in the transmitters. The GPS records the signals that it received. The times for each transmitter are subtracted from the time your GPS received them. These are recorded as t1, t2, and t3. The numbers correspond to the times from the respective transmitter locations. .

```
#13:  t1 := -0.8139056099
```

```
#14:  t2 := 0.7980845608
```

```
#15:  t3 := -0.4685532484
```

The actual times that it took to travel from the trasmitters to your GPS are t1+d, t2+d, and t3+d, where d is the time discrepency of your GPS. Thus, we have three unknown values: the x and y coordinates of your position and d, the time discrepency of your GPS.

Your distance from each of the transmitters is the velocity of the signal multiplied by the time that it took to travel to the GPS.

```
#16:  d1 := 200·(t1 + d)
```

```
#17:  d2 := 200·(t2 + d)
```

```
#18:  d3 := 200·(t3 + d)
```

But these distances are the radii of circles centered at each transmitter and passing through your position on the plane. We are now in the situation of Step 2 above. The one difference is that we now have three unknowns: x, y, and d. However we have three equations to solve. It may work out.

$$\#19:\quad (x - 75)^2 + (y - 150)^2 = d1^2$$

$$\#20:\quad (x - 510)^2 + (y - 110)^2 = d2^2$$

$$\#21:\quad (x - 150)^2 + (y - 20)^2 = d3^2$$

We simply use the Solve > System option as shown above and then approximate the result to obtain the following result. NOTE: the first result is not shown here. It is the exact symbollic solution to the system. It is rather messy since it involves expressions with sums of radicals and fractions of VERY large numbers. We want numbers that we can recognize. However, this will be a decimal approximation and we may loose some accuracy.

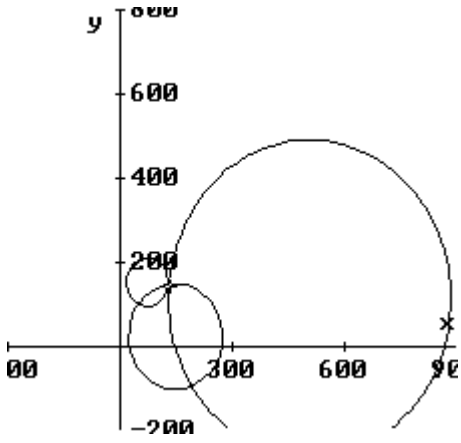
$$\#22:\quad \text{SOLVE}\left[(x - 75)^2 + (y - 150)^2 = d1^2,\ (x - 510)^2 + (y - 110)^2 = d2^2,\ (x - 150)^2 + (y - 20)^2 = d3^2\right], [x, y, d])$$

$$\#23:\quad [x = 452.5761121 \wedge y = 96.9671036 \wedge d = -1.092505996,\ x = 132 \wedge y = 145 \wedge d = 1.1]$$

Because we have three unknowns, we, once again, have two possible solutions to the system of equations. In this case we can rather easily eliminate one of them. Note that if d = -1.092505996 then each of the distances, d1, d2, and d3 will be negative. This is nonsense. We would have received the signals before they were sent. Thus, you can conclude that your position is at (132, 145) and the time discrepency is 1.1 second. Define d := 1.1 and graph expressions #19, #20, and #21.

NOTE: This can be done by holding down the <Ctrl> key and highlighting all three expressions followed by switching to the 2D-plot window and clicking on the plot button.

#24: define d:=1.1 by editing this line



After viewing the graph clear the value for d by typing d := . This action frees the symbol d so that it can be a variable.

```
#25:  clear d by editing this line to read d :=
```

It is time to try one on your own.

First clear the graphics screen by choosing the Edit > Delete All Plots from the 2D-plot menu.

Assume that you are working with a different GPS receiver, so you have no idea of the time difference between the transmitter clocks and your GPS clock. You are still in the same region of the plane as in the example. Here we go!

Exercise

6. Your GPS receives signals from three transmitters located at positions in the plane given as (320, 190), (50, 100), and (570, 50), respectively. The time differences recorded by your GPS receivers and the three transmitters are given as t1 := 3.113941029 seconds, t2 := 3.113941029 seconds, and t3 := 4.102775637 seconds, respectively. Enter these times by editing the lines below.

```
#26:  enter t1 by editing this line

#27:  enter t2 by editing this line

#28:  enter t3 by editing this line
```

Edit the next three lines to give the formulas for the distance from each of the three transmitters. Don't forget that you need to include the time discrepency between the transmitters and your GPS in these formulas.

```
#29:  enter the distance to transmitter #1 by editing this line

#30:  enter the distance to transmitter #2 by editing this line

#31:  enter the distance to transmitter #3 by editing this line
```

Now that you have the distances, you can enter the three equations that you will need to solve in order to determine your position.on the plane. Edit the three expressions below so that they contain the equations.

```
#32:  equation 1

#33:  equation 2

#34:  equation 3
```

After editing the above lines, solve the equations you entered using the Solve>Equations option. In this case, the distances for either of the answers are positive. However, we can eliminate one of the aswers rather easily. It lies outside of the region of the plane where we know we are located. Write the value for d, the time discrepancy, and plot the three circles.

```
#35:  value for d here
```

As a final exercise, you will demonstrate your understanding of the process by constructing a problem on your own for others to solve.

Exercise

7. Choose a set of 4 points that lie within the 2-D plot region of the plane that you have laid out and write down their coordinates. One of these points will be your position on the plane. The other three will be the locations of the three transmitters. Determine the distance of your point from each of the transmitters. Finally decide on a value for the variable, d, which is the time discrepency between the transmitters clocks and the clock on your GPS.
   a. Determine the time that it takes for each of the signals sent from the transmitters to reach your GPS.
   b. Using d, determine the values for t1, t2, and t3, the differences between the time sent by each transmitter and the time the signal is received.
   c. Write out your problem in a manner similar to the example and problem given above.
   d. Solve the problem and compare the answer with the coordinates for your position. This is a check that you correctly constructed the problem.

```
#36:  InputMode := Word
```

One more letter from Ernest Carpenter:

Ernest Carpenter (South Africa)

The article on Turtle Graphics on the TI-92, in DNL57, also brought back some nice memories when I first obtained my TI-92+ (before that I was actually a die hard HP48GX fan☺ and also used the QLIB referred to in the "Actuarial Math" article) and TG was one of the first programs I load onto it!

Now that I'm finished babbling, just a few quick questions to get your option on a matter or two:

- Do you think we will see a scaled down version of Cabri3D for the TI-89/92+/V200?
- Do you think from an education point of view whether the TI Graphing Calculators (or Graphing calculators in general TI,HP, Casio) would still be around for a while or will it disappear and make place for more windows based handhelds like the PocketPC or PalmOS devices? The main reason for this question is to see if developers or users in general should still focus on developing software/apps for the current devices or shift their efforts towards the new technology marked?
- I know that the use of technology in RSA, especially on a secondary (high school) mathematical level, is almost non-existent. How do the students/pupils in Europe benefit from using technology, like for instance the TI-92 or Derive?

Finally, any indication when you will 'publish' the articles "Tools for 3D-Problems" and "CAD-Design with DERIVE and the TI"?
(I know I'm persisted with the above mentioned question, but maybe my persistence would pay-off☺)

Thanks again for all the information and articles, I appreciate it!

Enjoy your day!

Regards
Ernest Carpenter

CADD Co-ordinator
Strategic Projects
De Beers(NM)

Titbits from Algebra and Number Theory (30)

by Johann Wiesenbauer, Vienna

Let's consider the following innocent looking problem: Given a positive integer n, is there a right triangle with rational sides a,b,c and area n? In other words, we are looking for rational solutions (a,b,c) of the system consisting of the two equations:

$$a^2+b^2=c^2, \quad \frac{ab}{2}=n \qquad (*)$$

Since (a,b,c) is a rational solution w.r.t. to the area n, if and only if (ta,tb,tc) is a rational solution w.r.t. to the area t^2n for any nonzero integer t, there is no loss of generality if we always assume that n is squarefree in the following. A squarefree positive integer n, for which the system of equations above has rational solutions, is called a *congruent number*.

Just to get a feeling for the problem, let's look now into some small cases. For example, are there rational solutions for $n = 5$ and $n = 6$, respectively? Well, this shouldn't be too hard for $n = 6$, as the "Egyptian triangle" with the sides $3,4,5$ clearly solves the problem. On the other hand, it might take you a while to see that $n = 5$ is a congruent number and e.g. $a = 3/2$, $b = 20/3$, $c = 41/6$ is a rational solution of (*) here. Just in case you wonder why I didn't start with the first three squarefree natural numbers, namely $n = 1, 2, 3$, the answer is very simple: There aren't any rational solutions in those cases, but this is far from being obvious. (E.g., the case $n = 1$ was settled by Fermat around 1650.)

There is a completely different way of looking at this problem, which will give us valuable insights though. Using Derive the following computations show that for a fixed n we can assign to each rational solution (a,b,c) of (*) a rational point (x,y) with $y \neq 0$ on the curve $y^2 = x^3 - n^2x$ and also the other way round.

$$\left[x := \frac{a\cdot(a - \sqrt{(a^2 + b^2)})}{2},\ y := \frac{a^2\cdot(\sqrt{(a^2 + b^2)} - a)}{2},\ n := \frac{a\cdot b}{2}\right]$$

$$y^2 - x^3 + n^2\cdot x = 0$$

$$[x :=,\ n :=]$$

$$\left[y := \pm\sqrt{(x^3 - n^2\cdot x)},\ a := \left|\frac{x^2 - n^2}{y}\right|,\ b := \left|\frac{2\cdot n\cdot x}{y}\right|,\ c := \left|\frac{x^2 + n^2}{y}\right|\right]$$

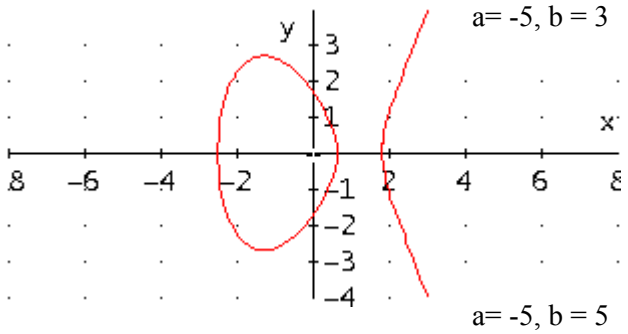
$$\left[a^2 + b^2 - c^2,\ \frac{a\cdot b}{2}\right] = [0,\ |n|]$$

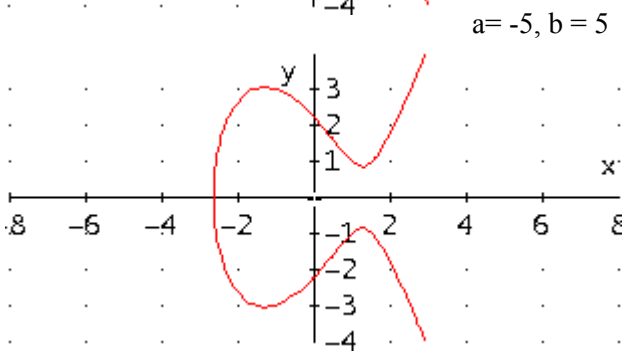
Algebraic curves of the form $y^2 = f(x)$, where $f(x)$ is a polynomial over some field F without multiple roots, are called elliptic curves. In many applications, it suffices to consider elliptic curves of the special form

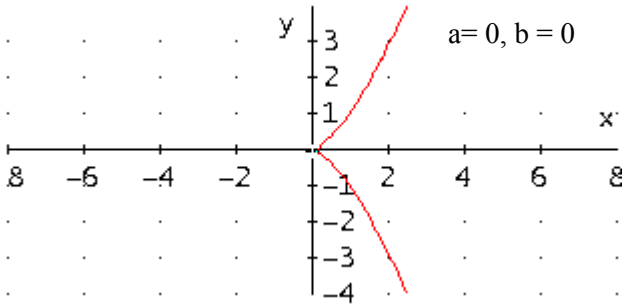
$$y^2 = x^3 + ax + b \quad (a,b \in F) \qquad (**)$$

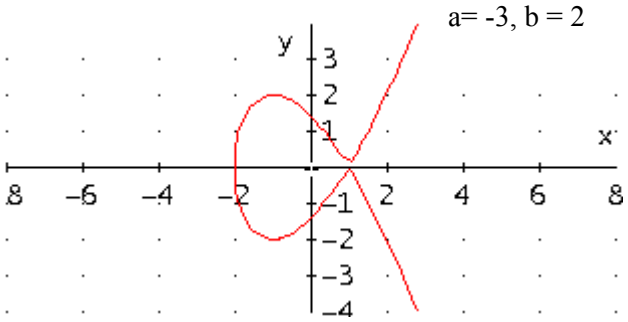
where the condition that the polynomial on the right-hand side has no multiple roots becomes the condition that its so-called discriminant $\Delta = 4a^3 + 27b^2$ doesn't vanish in F.

In the special case $F=\mathbb{R}$ we can take advantage of sliderbars for the parameters a and b (e.g. in the range $[-10,10]$) to produce the following pictures:







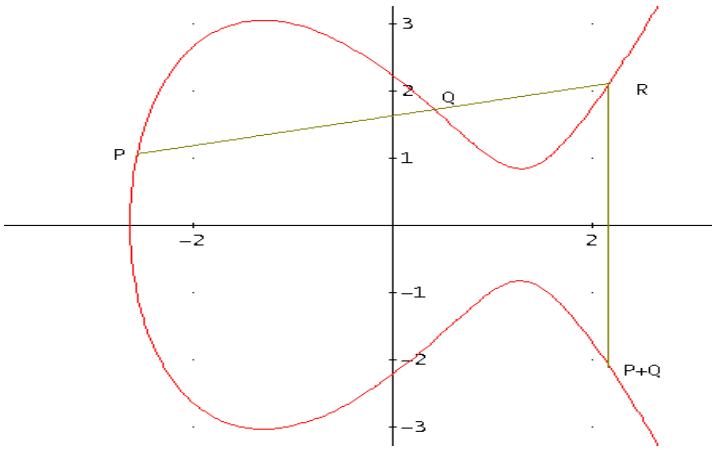


Note that only the first two curves are elliptic curves, while the other two are not due to singularities.

Maybe you have already heard of elliptic curves and their usefulness, e.g. in solving Diophantine equations (the most notable example is A.Wiles' proof of FLT!) and in encrypting messages. Hence, unless you are already an expert in this field, you might wonder: What is so special about them?

Well, there are many possible answers, but one of the most outstanding features of elliptic curves is certainly the fact that it is possible to define an addition on their points which turns them into abelian groups.

Considering again the field $F = \mathbb{R}$, the basic procedure to obtain the sum of two points P and Q is the following: Draw the chord through P and Q, find its intersection point R with the curve and define $P+Q$ as the mirror image of R w.r.t. the x-axis, as shown in the picture below.



The rough description above must be supplemented though in the case, where P and Q have the same x-coordinate. Then either P and Q coincide, in which case the chord becomes a tangent or/and P and Q are mirror images to each other w.r.t. the x-axis. In the latter case, we use the usual convention from projective geometry that the intersection point R with the curve (and hence also $P+Q$) is the point O at infinity. In particular, this point O has to be added to the points of the elliptic curve and plays the important role of the neutral element for our addition.

Let's now use the informal description above to compute the coordinates of $P+Q$, where $P = (x_1, y_1)$ and $Q = (x_2, y_2)$, under the assumption that $P + Q \neq O$. Then

$$k = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1}, & \text{if } x_1 \neq x_2 \\ \dfrac{3x_1^2 + a}{2y_1}, & \text{if } x_1 = x_2, y_1 \neq 0 \end{cases}$$

is the slope of the chord or tangent, respectively, and after an easy computation we get for the coordinates of $P+Q=(x_3,y_3)$ the equations

$$x_3 = k^2 - x_1 - x_2, \quad y_3 = -y_1 + k(x_1 - x_3)$$

It is important to notice that these formulas, actually derived under the condition $F = \mathbf{R}$, also make sense for an arbitrary field F, and although there is no longer a geometrical interpretation then, it is still true that the set of points of an elliptic curve along with this addition is always an abelian group.

When trying to implement this addition (and, of course, this is the next point on our agenda!) we must somehow specify the field F. In order to avoid making things to complicated for a first approach (for example, by allowing arbitrary fields for which the basic operations are given by functions defined by the user), I will consider in the following only the case, where F is either a subfield of the field $\mathbb{C}$ of complex numbers or a residue class ring $\bmod p$ for some prime p. Then a routine for the addition of two points U and V on an elliptic curve given by an equation (**) could look like this:

```
inv(x, p) :=
  If p = 0
     1/x
     INVERSE_MOD(x, p)

add(u, v, a, p := 0, k_) :=
  Prog
    If u↓1 + v↓1 = ∞
       RETURN [1/(1/u↓1 + 1/v↓1), 1/(1/u↓2 + 1/v↓2)]
    If u↓1 = v↓1
       Prog
         If MOD(u↓2 + v↓2, p) = 0
            RETURN [∞, ∞]
         k_ := MOD((3·u↓1^2 + a)·inv(2·u↓2, p), p)
         If k_ = ?
            RETURN GCD(2·u↓2, p)
       Prog
         k_ := MOD((v↓2 - u↓2)·inv(v↓1 - u↓1, p), p)
         If k_ = ?
            RETURN GCD(v↓1 - u↓1, p)
    v := MOD(k_^2 - u↓1 - v↓1, p)
    [v, MOD(- u↓2 + k_·(u↓1 - v), p)]
```

A few comments might be in order here. First, you should know that in the case $p = 0$, i.e. if F is assumed to be a subfield of $\mathbf{C}$, those reductions $\bmod p$ will do no harm, although they are of course superfluous! Second, in the case $F = \mathbb{Z}_p$, we will occasionally p allow to be a composite number, which leads to an "elliptic pseudocurve", for which addition is sometimes undefined. In those cases, a non-trivial divisor is returned as the result of the addition, which could be just

the thing you were after.

For the time being, we're dealing with a more down-to-earth problem though, namely the determination of some points on an elliptic curve again given by an equation $y^2 = x^3 + ax + b$. If $F = \mathbb{Z}_p$, and p is not too large, we should be able to compute even all its points. As for other fields, we restrict ourselves here to the most important case $F = \mathbb{Q}$, where the following routine computes all points up to a certain user-defined "height" h, which is an upper bound for $|r|,|s|$ in a representation $x = \frac{r}{s}$ of the x-coordinate of a point (x,y) on the curve. At any rate, the point O at infinity is added.

```
points(a, b, p := 0, h := 100, s := ±1, h_ := 0, f_, u_ := {[∞, ∞]}, x_ := 0, y_) :=
  If p > 0
     Loop
       y_ := SQUARE_ROOT(x_^3 + a·x_ + b, p)
       If NUMBER?(y_)
          u_ := u_ ∪ {[x_, y_], [x_, MOD(-y_, p)]}
       x_ :+ 1
       If x_ = p
          RETURN u_
     Loop
       h_ :+ 1
       WRITE(h_)
       If h_ > h
          RETURN u_
       f_ := SELECT(GCD(n_, h_) = 1, n_, 0, h_)/h_
       Loop
         x_ := FIRST(f_)
         y_ := √(x_^3 + a·x_ + b)
         If RATIONAL?(y_)
            u_ := ADJOIN([x_, s·y_], u_)
         y_ := √(- x_^3 - a·x_ + b)
         If RATIONAL?(y_)
            u_ := ADJOIN([-x_, s·y_], u_)
         x_ := 1/x_
         y_ := √(x_^3 + a·x_ + b)
         If RATIONAL?(y_)
            u_ := ADJOIN([x_, s·y_], u_)
         y_ := √(- x_^3 - a·x_ + b)
         If RATIONAL?(y_)
            u_ := ADJOIN([-x_, s·y_], u_)
         f_ := REST(f_)
         If f_ = [] exit
```

Before applying this routine, one should make sure though that the parameters a and b are chosen appropriately, i.e. $\Delta = 4a^3 + 27b^2$ must not vanish in F, as mentioned above. This can be accomplished by

```
                                  3       2
regular?(a, b, p := 0) := MOD(4·a  + 27·b , p) ≠ 0
```

At last, we are ready to apply the tools we have gathered so far to our

introductory problem! (Note that all the curves $y^2 = x^3 - n^2x$ at issue are non-singular due to the nonzero discriminant $\Delta = 4a^3 + 27b^2 = -4n^6$ in $F = \mathbb{Q}$.) Okay, maybe still not quite, as it will be convenient to have a routine that translates a rational point $U \neq O$ on those curves with $y \neq 0$ into the corresponding right triangle with rational sides a,b,c.

```
triangle(u, n) := [|u_1^2 - n^2 / u_2|, |2·n·u_1 / u_2|, |u_1^2 + n^2 / u_2|]
```

```
points(-36, 0)
```

$$\left\{[-6, 0], [-3, \pm 9], [-2, \pm 8], \left[-\frac{6}{49}, \pm\frac{720}{343}\right], [0, 0], [6, 0], \left[\frac{25}{4}, \pm\frac{35}{8}\right], [12, \pm 36], [18, \pm 72], [\infty, \infty]\right\}$$

```
TABLE(triangle(u, 6), u, SELECT(u_2 ≠ 0 ∧ u_2 ≠ ∞, u, points(-36, 0)))
```

$$\begin{bmatrix} [-3, \pm 9] & [3, 4, 5] \\ [-2, \pm 8] & [4, 3, 5] \\ \left[-\frac{6}{49}, \pm\frac{720}{343}\right] & \left[\frac{120}{7}, \frac{7}{10}, \frac{1201}{70}\right] \\ \left[\frac{25}{4}, \pm\frac{35}{8}\right] & \left[\frac{7}{10}, \frac{120}{7}, \frac{1201}{70}\right] \\ [12, \pm 36] & [3, 4, 5] \\ [18, \pm 72] & [4, 3, 5] \end{bmatrix}$$

As you can see above for $n = 6$, by searching for rational points up to the default height $h=100$, we have got another (far less obvious!) right triangle with rational sides $a = 7/10$, $b = 120/7$, $c = 1201/70$ and area 6.

What about the other squarefree numbers n in the range $1 \le n \le 10$? Take a look at the following table with all the rational points found up to height 100:

```
TABLE(points(-n^2,0),n,SELECT(SQUAREFREE(n),n,1,10))
```

```
DisplayFormat:=Compressed
```

$$\begin{bmatrix} 1 & \{[-1,0],[0,0],[1,0],[\infty,\infty]\} \\ 2 & \{[-2,0],[0,0],[2,0],[\infty,\infty]\} \\ 3 & \{[-3,0],[0,0],[3,0],[\infty,\infty]\} \\ 5 & \left\{[-5,0],[-4,\pm 6],\left[-\frac{5}{9},\pm\frac{100}{27}\right],[0,0],[5,0],\left[\frac{25}{4},\pm\frac{75}{8}\right],[45,\pm 300],[\infty,\infty]\right\} \\ 6 & \left\{[-6,0],[-3,\pm 9],[-2,\pm 8],\left[-\frac{6}{49},\pm\frac{720}{343}\right],[0,0],[6,0],\left[\frac{25}{4},\pm\frac{35}{8}\right],[12,\pm 36],[18,\pm 72],[\infty,\infty]\right\} \\ 7 & \left\{[-7,0],\left[-\frac{63}{16},\pm\frac{735}{64}\right],\left[-\frac{49}{25},\pm\frac{1176}{125}\right],[0,0],[7,0],[25,\pm 120],[\infty,\infty]\right\} \\ 10 & \{[-10,0],[0,0],[10,0],[\infty,\infty]\} \end{bmatrix}$$

Well, I already told you that $1,2,3$ aren't congruent numbers and from the table we can conclude that not only 5 and 6, but also 7 is a congruent number for sure. But what about $n = 10$? It is true that we haven't found rational solutions of $y^2 = x^3 - 100x$ except for the trivial ones, which are of no use here, but the tantalizing question remains: Was this because our search limit was simply too low or can it be really proven that $n = 10$ is not a congruent number?

If this is what you asked yourself (and hopefully you did!) then I have good news and bad news for you. First the good news: There is a remarkably simple test by J.B.Tunnell (1983), in order to find out whether a squarefree positive integer is congruent or not. The bad news: Only "no" answers are provably reliable, while "yes" answers are not. It must be said though that no counterexample has been found so far, in fact, such a counterexample would contradict the famous conjecture by Birch and Swinnerton-Dyer (one of the 7 millennium problems, each worth $ 1,000,000 for the first solver), which is widely accepted by the community of mathematicians.

For this test, we must compute the numbers r and s of integer solutions of the equations $x^2 + 2ay^2 + 8z^2 = \frac{n}{a}$ and $x^2 + 2ay^2 + 32z^2 = \frac{n}{a}$, where $a = 2 - (n \bmod 2)$, and simply check whether $r = 2s$ or not. The following routine will do the trick. (By the way, if you feel like a programming challenge, I cordially invite you to have a try at this task yourself before looking at my solution below. Although it is not too hard in principle, one has to take care not to get lost in too many cases and subcases, resulting in ugly if-statements.)

```
congruent?(n, a_, u_, r_ := 0, s_ := 0, x_, y_, u_, v_) :=
  Prog
    a_ := 2 - MOD(n, 2)
    n :/ a_
    u_ := REST(SORT({0, ..., FLOOR(√n)}·{0, ..., FLOOR(√(n/(2·a_)))}))
    Loop
      If u_ = []
         RETURN r_ = 2·s_
      x_ := FIRST(FIRST(u_))
      y_ := FIRST(REST(FIRST(u_)))
      v_ := x_^2 + 2·a_·y_^2
      z_ := √((n - v_)/8)
      If INTEGER?(z_)
         Prog
           r_ :+ (2 - 0^x_)·(2 - 0^y_)·(2 - 0^z_)
           If EVEN?(z_)
              s_ :+ (2 - 0^x_)·(2 - 0^y_)·(2 - 0^z_)
      u_ := REST(u_)
```

As you can see from the table below, 10 is not contained among the numbers below 100, that passed the test, which is a proof that it isn't congruent, indeed.

```
TABLE(congruent?(n), n, SELECT(SQUAREFREE(n), n, 1, 100))'
```

1	2	3	5	6	7	10	11	13	14	15	17	19	21	22	23	26	29
false	false	false	true	true	true	false	false	true	true	true	false	false	true	true	true	false	true
30	31	33	34	35	37	38	39	41	42	43	46	47	51	53	55	57	58
true	true	false	true	false	true	true	true	true	false	false	true	true	false	true	true	false	false
59	61	62	65	66	67	69	70	71	73	74	77	78	79	82	83	85	86
false	true	true	true	false	false	true	true	true	false	false	true	true	true	false	false	true	true
87	89	91	93	94	95	97											
true	false	false	true	true	true	false											

From a closer look at the table you will also learn that all squarefree numbers of the form $8k+5$, $8k+6$ or $8k+7$ in the given range passed the test and hence are congruent numbers, as it has been proven by others that all results of Tunnell's criterion are correct, if $n < 1000$. In fact, congruent numbers not of this form are a minority, like $34, 41, 65$ in our example above.

Before coming to an end, let's also talk a little bit about the case, where $F = \mathbb{Z}_p$ for some prime p, which plays an important role in modern cryptography. One of the most important issues there is the computation of the number N_p of points on an elliptic curve E again given by the equation $y^2 = x^3 + ax + b$ $(a,b \in F)$. It turns out that the algorithms used for this computation are very similar to those introduced in my last column to solve the discrete logarithm problem (DLP) and it will soon become clear why.

A very first, though very crude implementation, would make use of our routine `points(a,b,p)` and simply count the number of resulting points, i.e.

```
Np(a, b, p) := DIM(points(a, b, p))
```

As the following simple example shows this is far too slow though.

```
regular?(1, 1, 1009) = true
```

```
Np(1, 1, 1009) = 1034   (18.4s )
```

A better idea is to compute for every x in the range $\{0,1,\ldots,p-1\}$ not the points with this x-coordinate, but only the number of possible y-values, i.e. the number of solutions of $y^2 \equiv x^3 + ax + b \mod p$. According to Euler's criterion from the theory of quadratic residues this number is given by $1+((x^3+ax+b)^{(p-1)/2} \bmod p)$, which yields the implementation

```
                                    3                 (p - 1)/2
Np(a, b, p) := p + 1 + Σ(MODS((x  + a·x + b)                  , p), x, 0, p - 1, 1)
```

```
Np(1, 1, 1009) = 1034   (0.03s !!)
```

On the other hand, the following computation

```
                          6
Np(1, 1, NEXT_PRIME(10 )) = 1000727
```

still took 156.5s on my PC! It looks as if we are not yet arrived at a method for counting points on elliptic curves $\bmod p$, where p has say 10-20 digits!

Hence, let's use some results from the theory of elliptic curves, which may bring us further. In the first place, there is the well-known theorem by Hasse, which states that the number of points of an elliptic curve $E \bmod p$, which we denote by $\#E_p$ in the following, always lies in a relatively close vicinity of $p + 1$. To be more precise, the absolute difference $\left|\#E_p - (p+1)\right|$ can never get larger than $2\sqrt{p}$. How can we exploit this feature for our purposes?

Well, as you might know, the order of an element of a finite group is always a divisor of the group order. This means that starting with an arbitrary point U on the elliptic curve E_p, there is always some multiple kU with k "nearby" $p + 1$, such that $kU = O$. If the order of U is "big enough", more precisely not smaller than the length $4\sqrt{p}$ of the "Hasse interval" $[p+1-2\sqrt{p}, p+1+2\sqrt{p}]$, then this k must be $\#E_p$ itself, as it is the only candidate.

Okay, let's do the programming now. In the first place, we need a routine that computes for any integer n the multiples nU of a point U on the elliptic curve E_p, which is given as usually by an equation $y^2 = x^3 + ax + b$. Again, try to write this routine yourself, before looking at my solution below.

```
mult(u, n, a, p, s_ := [∞, ∞]) :=
  If n < 0
     mult([u↓1, - u↓2], -n, a, p)
     Loop
       If n = 0
          RETURN s_
       If ODD?(n)
          s_ := add(s_, u, a, p)
       u := add(u, u, a, p)
       n := FLOOR(n, 2)
```

We will later have many opportunities to see the "power" of this small jewel of programming, hence I'll skip examples here.

The next thing we must do is to solve the equation $kU = O$ for a given point U, where the integer variable is the unknown here. This is where our algorithms for the DLP come into play. (Hopefully, you can see this yourself, although the operation is written additively, whereas I used the multiplicative notation to introduce the DLP in my last column!) For example, by slightly adapting the

baby-step giant-step algorithm we get the following routine. (Note that the variables u, a, p have the same meaning as above!)

```
BSGS(u, a, p, j_, i_ := 0, r_, s_, t_, v_, x_) :=
  Prog
    r_ := CEILING(p^(1/4))
    s_ := mult(u, p + 1 - 2·r_^2, a, p)
    t_ := mult(u, 2·r_, a, p)
    v_ := ITERATES(add(u, w_, a, p), w_, [∞, ∞], r_)
    x_ := v_ COL 1
    Loop
      If MEMBER?(FIRST(s_), x_)
         Prog
           j_ := POSITION(FIRST(s_), x_) - 1
           If s_ = v_↓(j_ + 1)
              j_ := -j_
           RETURN p + 1 + 2·(i_ - r_)·r_ + j_
      s_ := add(s_, t_, a, p)
      i_ :+ 1
```

Finally, to get the order of U, we must successively remove from the returned k all primes q for which still $\frac{k}{q}U = O$ holds.

```
ord(u, a, p, o_, q_) :=
  Prog
    o_ := BSGS(u, a, p)
    q_ := (FACTORS(o_)) COL 1
    Loop
      If q_ = []
         RETURN o_
      Loop
        If mult(u, o_/FIRST(q_), a, p) ≠ [∞, ∞] exit
        o_ :/ FIRST(q_)
        If MOD(o_, FIRST(q_)) > 0 exit
      q_ := REST(q_)
```

Well, are you ready for the most advanced `Np(a,b,p)` so far? Here you are!

```
Np(a, b, p, g_ := 1, h_, s_ := 1, x_, y_) :=
  Prog
    If p ≤ 229
       RETURN p + 1 + Σ(MODS((x^3 + a·x + b)^((p - 1)/2), p), x, 0, p - 1, 1)
    Loop
      g_ :+ 1
      If JACOBI(g_, p) = -1 exit
    Loop
      Loop
        x_ := RANDOM(p)
        s_ := JACOBI(x_^3 + a·x_ + b, p)
        If s_ = ±1 exit
      h_ := g_^((1 - s_)/2)
      y_ := SQUARE_ROOT(h_·(x_^3 + a·x_ + b), p)
      h_ := ord([MOD(h_·x_, p), MOD(h_·y_, p)], MOD(h_^2·a, p), p)
      If h_ > 4·√p
         RETURN (p + 1)·(1 - s_) + s_·CEILING(p + 1 - 2·√p, h_)·h_
```

For those of you, who, who wonder how this works in detail, here comes a little support. At first, you should be able to see that $s\_ = 1$ means that for a randomely chosen x in $\{0,1,..,p-1\}$ there exists a point (x,y) on the curve indeed. In the opposite case $s\_ = -1$, there is still no need to despair, as every curve E_p has a "twin brother" E'_p, called its *quadratic twist*, with the property that there is a point with a given x-coordinate on E_p if and only if there is no such point on E'_p. Its equation is given by

$$y^2 = x^3 + ag^2x + bg^3$$

where g is some quadratic nonresidue $\bmod p$. From this easily follows that

$$\#E_p + \#E'_p = 2(p+1).$$

Hence, if we know one of the two summands on the left, we know also the other one! This will do for all primes $p>229$ as noted by Mestre. If this condition is not fulfilled, then we simply use one of the previous formulas, which are fast enough for those small primes!

Okay, I have to come to an end, as Josef is already tearing out his hair, both because of the length of this paper and its late delivery. Hence, let me conclude with some small examples for the routines above. First, let's take a really huge number k, say the $\mathrm{lcm}(1,2,..,30000)$, to form the multiple kU for $U = (1,1) \bmod F_7$. (After all, Fermat told us that F_7 is a prime, didn't he?)

```
DIM(LCM([1, ..., 30000])) = 13013

                                                    7
mult([1, 1], LCM([1, ..., 30000]), 139, 2^(2^7) + 1) = 59649589127497217 (22.7s !!)
```

Whew, what's that? Obviously Fermat was mistaken and our routine has reported a 17-digit prime factor p of his number. And now the acid test for our third try at `Np(a,b,p)`, which is the time-consuming part of `ord(u,a,p)`!

```
ord([1, 1], 139, 59649589127497217) = 29824794689700405  (106.7s)

FACTOR(29824794689700405) = 3·5·37·419·1021·4783·26263
```

It worked! And what's more, it worked in a reasonable time! What you saw above is the basic principle of the celebrated ECM (=Elliptic Curve Method), introduced by H.Lenstra Jr. in 1987. As you see, by a fortunate coincidence the order of the point $(1,1)$ on the elliptic curve $E_{29824794689700405}$ splits up into many small prime factors, the largest being 26263, which is smaller than our bound 30000 above. Actually, without this prime factor, the order would be even "smooth" w.r.t. to the much lower bound 4783 (or say 5000). In an actual

implementation of the ECM, one takes advantage of this fact by introducing two bounds B_1 and B_2, where B_1 covers all prime factors with one possible exception whereas B_2 (usually 10 to 100 times larger than B_1) covers also this one-off.

As for the following Derive-implementation, here is the meaning of the variables: n is the number to factor, U is a point on the elliptic curve used, a is the coefficient in the equation of the curve, and s and t are the two bounds above. (Note that, as many times before, again the seemingly missing coefficient b of the curve equation is determined implicitly by the coordinates of the point U!)

```
ECM(n, u, a, s, t, l_ := [], m_ := 0, u_, p_ := 2, q_) :=
  Prog
    Loop
      u := mult(u, p_^FLOOR(LN(s), LN(p_)), a, n)
      If NUMBER?(u)
         RETURN u
      p_ := NEXT_PRIME(p_)
      If p_^2 > s exit
    Loop
      u := mult(u, p_, a, n)
      If NUMBER?(u)
         RETURN u
      p_ := NEXT_PRIME(p_)
      If p_ > s exit
    u_ := mult(u, p_, a, n)
    q_ := NEXT_PRIME(p_)
    Loop
      If NUMBER?(u_)
         RETURN u_
      If p_ > t
         RETURN 1
      Loop
        If q_ - p_ ≤ m_ exit
        m_ :+ 2
        v_ := mult(u, m_, a, n)
        If NUMBER(v_)
           RETURN v_
        l_ := APPEND(l_, [v_])
      u_ := add(u_, l_↓((q_ - p_)/2), a, n)
      p_ := q_
      q_ := NEXT_PRIME(q_)
```

Will this be notably faster than our crude computation above? Sure!

$$ECM\left(\begin{matrix}7\\2\\2\end{matrix} + 1, [1, 1], 139, 5000, 30000\right) = 59649589127497217 \quad (4.89s\ !)$$

Don't take this time as the total running time for ECM though, as there are quite a few failures until you find the "appropriate" value $a = 139$ above. Okay, okay, let's stop here ...

(As always, questions or remarks to j.wiesenbauer@tuwien.ac.at)

Recommended Readings

Liebe Kolleginnen und Kollegen
Hansruedi Vollmer und ich haben beschlossen, die 2. ueberarbeitete und verbesserte Auflage unseres Buches "Skalarprodukte, Schwingungen, Signale" auf dem Bildungsserver

www.swisseduc.ch

kostenlos anzubieten. Bitte weitersagen.

Mit den besten Gruessen
Hansruedi Vollmer und Hansruedi Schneebeli

Hansruedi Vollemer and Hansruedi Schneebeli offer a free download of their book “Scalar Products, Oscillations, Signals” (in German) from www.swisseduc.ch.

Information provided by Guido Herweyers, Belgium

You can find the pdf-version of “**A CASE FOR CAS**” online at

http://www.uhasselt.be/scholennetwerk/ig/CAS/CAS.html
(from www.t3vlaanderen.be → publicaties)

This is a T^3-Europe product written by Ian Forbes, Gert Schomacker, Guido Herweyers, René Hugelshofer and Josef Böhm.

Information provided by Robert McCollum, USA

This is just a reminder to send in a proposal to speak at the NCTM Chicago Regional to be held in Chicago Sept. 20-22, 2006 at the Hyatt McCormick Place. The deadline for submissions is JULY 11, 2005 (this summer!)

I would love to see you sharing your expertise with CAS at this conference.

The URL is www.nctm.org/speak/

They are particularly looking for CAS speakers but most anything you submit would be appreciated.

Take Care,
Bob